

Decidable Models of Recursive Asynchronous Concurrency

Jonathan Kochems

C.-H. Luke Ong

Abstract—Asynchronously communicating pushdown systems (ACPS) that satisfy the empty-stack constraint (a pushdown process may receive only when its stack is empty) are a popular decidable model for recursive programs with asynchronous atomic procedure calls. We study a relaxation of the empty-stack constraint for ACPS that permits concurrency and communication actions at any stack height, called the *shaped stack* constraint, thus enabling a larger class of concurrent programs to be modelled. We establish a close connection between ACPS with shaped stacks and a novel extension of Petri nets: *Nets with Nested Coloured Tokens* (NNCTs). Tokens in NNCTs are of two types: *simple* and *complex*. Complex tokens carry an arbitrary number of coloured tokens. The rules of NNCT can synchronise complex and simple tokens, inject coloured tokens into a complex token, and eject all tokens of a specified set of colours to predefined places. We show that the coverability problem for NNCTs is TOWER-complete. To our knowledge, NNCT is the first extension of Petri nets—in the class of nets with an infinite set of token types—that is proven to have primitive recursive coverability. This result implies TOWER-completeness of coverability for ACPS with shaped stacks.

I. INTRODUCTION

In recent years the study of decision procedures for concurrent pushdown systems has proved immensely fruitful. Substantial advances have been made in the algorithmic verification of *asynchronous programs*, i.e. recursive programs with asynchronous atomic procedure calls [42, 27, 22]. Asynchronous programs can be modelled naturally by *asynchronously communicating pushdown systems* (ACPS)—a dynamic network of concurrent pushdown systems that communicate via a fixed, finite set of unbounded and unordered channels—subject to the “empty-stack restriction”, which means that a pushdown process cannot receive messages unless its stack is empty.

The empty-stack restriction prohibits arbitrary synchronisations between processes, thus ruling out classes of interesting programs for analysis by ACPS. For example, the server program in Figure 1 gives rise to an ACPS. The program spawns two processes, one running *server*, the other *despatcher*. The *server* process posts tasks to the channel *task_bag* which are continually removed and executed by the *despatcher* process (possibly posting further tasks to *task_bag* or spawning new processes). The *despatcher* process non-deterministically chooses to wait for a *stop* message from the *server*, or calls itself recursively. An interesting question for this program is whether the messages *ready* and *despatcher_done* can erroneously reside in the channels *task_bag* and *system* resp. at the same time. Such a question may be formulated as a *coverability* problem: is it possible to reach a configuration s that *covers* a configuration s_{cov} i.e. $s_{\text{cov}} \leq s$ where \leq is a preorder on the configuration-space. Unfortunately coverability is undecidable for ACPS in general [38], however it is decidable for ACPS

```

1  %%% Server
2  server() →
3    init_despatcher(), do_server(), post_task(),
4    case (*) of
5      true → server();
6      false → system ? stop
7    end,
8    task_bag ! stop.
9
10 post_task() → task_bag ! task, task_bag ? ok.
11
12 init_despatcher() → task_bag ! init, task_bag ? ready.
13
14 despatcher() →
15   task_bag ? init, task_bag ! ready,
16   task_bag ? task, task_bag ! ok,
17   do_task(),
18   case (*) of
19     true → despatcher();
20     false → task_bag ? stop, system ! despatcher_done end.
21
22 main() → spawn(server), spawn(despatcher), system ! stop.

```

Figure 1: Server in asynchronous programming style. The procedures *do_server* and *do_task* may be arbitrary terminating recursive procedures that are allowed to send messages and spawn new processes.

with the empty-stack restriction. Notice that the *server* process increases its call-stack at every recursive call while executing receives and sends: it does not satisfy the empty-stack restriction. Is it possible to relax the empty-stack restriction on ACPS while preserving the decidability of coverability? Fortunately, the answer is yes. In previous work [29], we introduced *asynchronous partially commutative pushdown systems* (APCPS), a model of recursive asynchronous concurrency in the form of a class of (partially commutative) *context free grammar*. An APCPS process is an equivalence class of words (over a set of non-terminal and terminal symbols) that allow the commutation of certain commutative non-terminals; the terminals determine the concurrency and communication side-effects, and the transition relation is essentially the leftmost derivation of the grammar. Intuitively, a non-terminal (or procedure) is *non-commutative* just if a “blocking operation”, such as receive, may be invoked when running it. In the APCPS setting, the empty-stack restriction is replaced by the *K-shape constraint* where $K \geq 0$, which limits the number of *non-commutative non-terminals* that may occur in any reachable process to no more than K . Processes may perform concurrent actions, message-send / receive or spawns at any time, provided all reachable processes fit the *K-shape* constraint. Consider a receive transition of the *server* process:

$$\begin{aligned}
 & (\text{task_bag} ? \text{ok}, \text{do_server}() \cdot \text{post_task}() \cdot L_4 \cdot L_8 \cdots L_8) \parallel \cdots \\
 & \xrightarrow{\text{task_bag} ? \text{ok}} (\text{do_server}(), \text{post_task}() \cdot L_4 \cdot L_8 \cdots L_8) \parallel \cdots
 \end{aligned}$$

where non-commutative and commutative procedures are marked in **red** and **green** resp. This transition is impossible

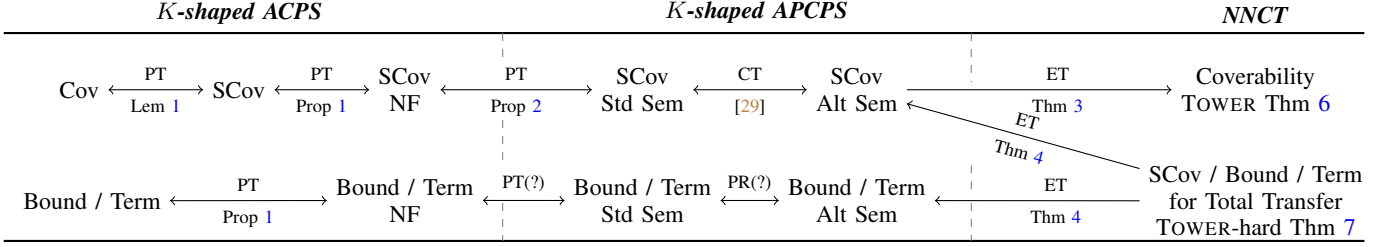


Figure 2: A “map” of this paper’s results. “ $P_1 \xrightarrow{\theta} P_2$ ” means that (decision problem) P_1 θ -reduces to P_2 . If the K -shaped is dropped the ACPS and APCPS part of the graph remains valid. *Legend:* PT = polynomial time, CT = constant time, ET = exponential time, PR = primitive recursive time, NF = normal form, $Std Sem$ = standard semantics, $Alt Sem$ = alternative semantics, $(?)$ = conjecture, Cov = coverability, $SCov$ = simple coverability, $Bound$ = boundedness, and $Term$ = termination.

under the empty-stack constraint; to satisfy the latter the server process is forced to empty its call-stack before it can make a receive transition: remembering to execute `do_server()`, `post_task()` and to return to line 4 (L_4) and eventually to perform all recursive executions of line 8 (L_8) is not possible. The state of the server process in this transition is representative; they always fit a suffix of the shape $(q, \blacksquare \blacksquare^* \blacksquare \blacksquare^* \blacksquare \blacksquare^*)$. This is in fact true for all processes in the program in Figure 1; thus the program fits the shape constraint. In general, the K -shaped constraint allows a pushdown process to remember *infinite state* information along a receive transition, whereas the empty-stack constraint limits it to be *finite state*.

The main result of *op. cit.* is that coverability is decidable for APCPS that satisfy the K -shape constraint. Though “semantic” in nature, the shape constraint follows from an easy-to-check syntactic condition [29] which seems natural and readily satisfied by recursive programs humans write.

Our contributions: The APCPS model is a hybrid model. On the one hand, it has the form of a partially commutative context-free grammar (in the sense of [9]) equipped with an operational semantics that specifies the behaviour of the concurrency and communication side-effects, such as send, receive and spawn. On the other, an APCPS determines a transition system which is very similar to that of an ACPS (the main difference is that APCPS processes are defined modulo a commutation relation). Our first contribution is to clarify the connections between APCPS and the standard and much studied ACPS. We show that there is a corresponding K -shape constraint for ACPS, which limits the number of non-commutative stack symbols that may occur in the “reachable” stacks. We prove that coverability for K -shaped ACPS is polynomial-time inter-reducible with (a simplified version of) coverability for K -shaped APCPS, which is decidable [29]; see Figure 2. Notice that the K -shaped ACPS model strictly extends the ACPS model with empty-stack restriction; in fact, the latter satisfies the 1-shape constraint by definition.

What is the complexity of coverability for K -shaped ACPS? We know that ACPS satisfying the empty-stack restriction are closely related to Petri nets: for example, their respective coverability problems are inter-reducible [22]. However the K -shape constraint captures a larger class of models than the empty-stack restriction. Is there an extension of Petri

nets that corresponds to K -shaped ACPS? Our second, major, contribution, are answers to these questions. (See Figure 2 for an overview of the technical results.)

(i) We introduce a non-trivial extension of Petri nets: *nested nets with coloured tokens* (NNCT). As the name suggests, NNCT feature, in addition to ordinary tokens (called *simple*), *complex tokens* that carry *coloured tokens*. Transitions may inject coloured tokens into a complex token; or transfer certain coloured tokens—those whose colour is from a specified set of *active colours*—from a complex token to predefined places. (ii) We show that coverability for NNCT is EXPTIME inter-reducible with simple coverability for APCPS (via the alternative semantics), and hence also inter-reducible with coverability for K -shaped ACPS.

(iii) We prove that coverability for NNCT is TOWER-complete, in the sense of Schmitz [40]. To our knowledge, NNCT is the first extension of Petri nets—in the class of nets with an infinite set of token types—that is proven to have primitive recursive coverability. To prove TOWER-membership of coverability for NNCT, we devise a geometrically inspired version of the Rackoff technique [37], which was originally used to prove the EXPSPACE-coverability for Petri nets. We obtain TOWER-hardness of coverability, boundedness and termination by adapting Stockmeyer’s ruler construction [43] to NNCT. We also establish the decidability of *boundedness* and *termination* for NNCT. Transferring our complexity analysis on NNCT implies, surprisingly, that the bound K on the number of “non-commutative procedure calls” in the shaped stack constraint is not the expensive resource. In fact, K influences only the number of colours n_{col} and complex places n_c of the simulating NNCT \mathcal{N} ; coverability is then decidable in space bounded by an exponential tower of height $O(n_s + \text{slog}(n_s \cdot n_{col} \cdot n_c))$ where n_s is the number of simple places of \mathcal{N} which is independent of K .

Notation: Let us write $\mathbb{N}^\infty = \mathbb{N} \cup \{\infty\}$, $\langle n \rangle = \{1, \dots, n\}$, and $\mathbb{M}[U]$ for the set of multisets over U . Explicit multisets are enclosed in $[\cdot]$; e.g. we write $[u, u, v^2]$ for the multiset containing two occurrences each of u and v . We write \emptyset for both the empty set and the empty multiset. We say that u is an element of the multiset $M \in \mathbb{M}[U]$, written $u \in M$, if $M(u) \geq 1$. If $M_1, M_2 \in \mathbb{M}[U]$, we write $M_1 \oplus M_2$ for the multiset union of M_1 and M_2 . If $M = M_1 \oplus M_2$ then

we define $M_1 = M \ominus M_2$. Let $M \in \mathbb{M}[U]$ and $U_0 \subseteq U$. We define $M \upharpoonright U_0$ to be the multiset M restricted to U_0 i.e. $(M \upharpoonright U_0) : u \mapsto M(u)$ if $u \in U_0$, and 0 otherwise. We write U^* for the set of finite sequences over U , i.e. maps from $\mathbb{N} \rightarrow U$, let β, γ, \dots range over U^* and we denote the *length* of sequence β by $|\beta|$. We define the *Parikh image* of $\beta \in U^*$ by $\mathbb{M}(\beta) : u \mapsto |\{i \mid \beta(i) = u\}|$. Let (U, \leq_U) be a preordered set; we extend \leq_U to $\mathbb{M}[U]$ and U^* as usual: (i) $M_1 \leq_{\mathbb{M}[U]} M_2$ just if $M_1 = [u_1, \dots, u_n]$, $M_2 = [u'_1, \dots, u'_m]$ and there is an injective map $h : \langle n \rangle \rightarrow \langle m \rangle$ such that for all $i \in \langle n \rangle$ we have $u_i \leq_U u'_{h(i)}$; (ii) $\beta \leq_{U^*} \beta'$ if there is a strictly monotone function $h : \langle |\beta| \rangle \rightarrow \langle |\beta'| \rangle$ and $\beta(i) \leq_U \beta'(h(i))$ for all $i \in \langle |\beta| \rangle$. We write $U_1 + U_2$ for the disjoint union of sets U_1 and U_2 . Let $f_1 : U_1 \rightarrow V_1$ and $f_2 : U_2 \rightarrow V_2$ be maps, then we define the map $f_1 + f_2 : U_1 + U_2 \rightarrow V_1 + V_2$ by $(f_1 + f_2)(in_i(u)) := in_i(f_i(u))$, for $i \in \{1, 2\}$, where in_i is the canonical injection of U_i into $U_1 + U_2$. Henceforth, to save writing, we elide in_i . We extend the operators \oplus, \ominus to functions, i.e., if $h_1, h_2 : V \rightarrow \mathbb{M}[U]$ then $(h_1 \oplus h_2)(v) := h_1(v) \oplus h_2(v)$ and $(h_1 \ominus h_2)(v) := h_1(v) \ominus h_2(v)$. For all sets U, V we define the map $\mathbf{0} : U \rightarrow \mathbb{M}[V]$ by $\mathbf{0}(u) = \emptyset$ for all $u \in U$. We write $f[u_1 \mapsto u'_1, \dots, u_n \mapsto u'_n]$ (we omit f if $f = \mathbf{0}$) for the function f' such that $f'(u) = f(u)$ for $u \neq u_i$, and $f'(u_i) = u'_i$ for all $i \in \langle n \rangle$.

II. RECURSIVE ASYNCHRONOUS CONCURRENCY

ACPS are prevalent as concurrent systems and their algorithmic verification is a central problem in verification. A variety of verification problems for asynchronous programs, and thus ACPS satisfying the empty-stack constraint, are polynomial-time inter-reducible to decision problems on Petri nets [22]. Due to this connection, we know that e.g. verification of safety properties is EXPSpace-complete. It is our goal to exhibit a similar connection between ACPS satisfying the shape constraint and an extension of Petri nets.

Definition 1. An *asynchronously communicating pushdown system* (ACPS) \mathcal{P} is a quintuple $\mathcal{P} = (\mathcal{Q}, \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R})$ composed of *control states* \mathcal{Q} , a *stack alphabet* \mathcal{A} , *channel names* Chan , *messages* Msg and *rules* \mathcal{R} , all finite sets; \mathcal{R} is a subset of $\mathcal{Q} \times \mathcal{A}^* \times \Lambda \times \mathcal{Q} \times \mathcal{A}^*$ where Λ is the set of *communication side-effects*: $\Lambda := \{c ? m, c ! m, \nu(q, \beta) : c \in \text{Chan}, m \in \text{Msg}, q \in \mathcal{Q}, \beta \in \mathcal{A}^*\} \cup \{\epsilon\}$.

We use the notation $(q, \beta) \xrightarrow{\lambda} (q', \beta')$ for rules. An action $\lambda = c ! m$ denotes the sending of the message m to channel c , $c ? m$ denotes the retrieval of m from c , and $\nu(q, \beta)$ denotes the spawning of a new process in state (q, β) . An ACPS \mathcal{P} gives rise to an infinite state transition system over $\mathbb{M}[\mathcal{Q} \times \mathcal{A}^*] \times (\text{Chan} \rightarrow \mathbb{M}[\text{Msg}])$. For simplicity, we write a configuration (say) $[(q, \beta), (q', \beta')], \{c_1 \mapsto [m_a, m_b, m_b], c_2 \mapsto []\}$ as $(q, \beta) \parallel (q', \beta') \triangleleft [m_a, m_b, m_b]^{c_1}, []^{c_2}$. We abbreviate a set of processes running in parallel as Π and a set of channels by Γ with names in Chan . The transition relation $\rightarrow_{\mathcal{P}}$ is defined as follows: suppose $(q, \beta) \xrightarrow{\lambda} (q', \beta') \in \mathcal{R}$ and $\beta_0 \in \mathcal{A}^*$ then

$$(q, \beta\beta_0) \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} (q', \beta'\beta_0) \parallel \Pi \triangleleft \Gamma \quad (\lambda = \epsilon)$$

$$\begin{aligned} (q, \beta\beta_0) \parallel \Pi \triangleleft \Gamma &\rightarrow_{\mathcal{P}} (q', \beta'\beta_0) \parallel \Pi \triangleleft \Gamma \oplus [c \mapsto [m]] & (\lambda = c ! m) \\ (q, \beta\beta_0) \parallel \Pi \triangleleft \Gamma \oplus [c \mapsto [m]] &\rightarrow_{\mathcal{P}} (q', \beta'\beta_0) \parallel \Pi \triangleleft \Gamma & (\lambda = c ? m) \\ (q, \beta\beta_0) \parallel \Pi \triangleleft \Gamma &\rightarrow_{\mathcal{P}} (q', \beta'\beta_0) \parallel (q_0, \beta_1) \parallel \Pi \triangleleft \Gamma & (\lambda = \nu(q_0, \beta_1)) \end{aligned}$$

Many interesting verification problems can be expressed in terms of transition systems that are endowed with a preorder on the state space which we call *pre-structured transition systems* (PSTS). Formally, a PSTS is a triple $\mathcal{S} = (S, \rightarrow_{\mathcal{S}}, \leq_{\mathcal{S}})$ such that $\leq_{\mathcal{S}}$ is a preorder on S , $\rightarrow_{\mathcal{S}} \subseteq S \times S$ a transition relation and we denote its transitive closure as $\rightarrow_{\mathcal{S}}^*$.

Definition 2. Let $\mathcal{S} = (S, \rightarrow_{\mathcal{S}}, \leq_{\mathcal{S}})$ be a PSTS and $s_0, s_{\text{cov}} \in S$ be configurations. The triple $Q = (\mathcal{S}, s_0, s_{\text{cov}})$ is a *coverability query* which is said to be a *yes-instance for coverability* if there is a reachable configuration s' that covers s_{cov} , i.e., $s_0 \rightarrow_{\mathcal{S}}^* s'$ and $s_{\text{cov}} \leq_{\mathcal{S}} s'$. The *boundedness problem* asks whether the set $\{s : s_0 \rightarrow_{\mathcal{S}}^* s\}$ is finite and the *termination problem* is to decide whether there exists an infinite path from s_0 in \mathcal{S} .

We augment ACPS with an order to yield a PSTS as follows: we order two processes $(q, \beta) \leq_{\mathcal{Q} \times \mathcal{A}^*} (q', \beta')$ just if $q = q'$ and either $\beta = \epsilon$ or $\beta = A \cdot \beta_0$, $\beta' = A \cdot \beta'_0$ and $\beta_0 \leq_{\mathcal{A}^*} \beta'_0$. Configurations are then ordered using the usual multiset and function extension: $\Pi \triangleleft \Gamma \leq_{\text{ACPS}} \Pi' \triangleleft \Gamma'$ just if $\Pi \leq_{\mathbb{M}[\mathcal{Q} \times \mathcal{A}^*]} \Pi'$ and if $\Gamma(c) \leq_{\mathbb{M}[\text{Msg}]} \Gamma'(c)$ for all $c \in \text{Chan}$.

An ACPS process $(q, A\beta)$ may process its own stack, test whether $\beta \geq_{\mathcal{A}^*} \beta_{\text{cov}}$, and record this in its local state q . We may thus assume that in a coverability query $(\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ all processes (q, β) of Π and Π_0 satisfy $\beta \in \{\epsilon, A \in \mathcal{A}\}$: we call such a query *simple* and the coverability problem restricted to simple queries *simple coverability*.

Lemma 1. Coverability and simple coverability for ACPS polynomial-time inter-reduce.

Control-states may be encoded in an enlarged stack-alphabet and hence we may w.l.o.g. consider ACPS in *normal form*: for all $(q, \beta) \xrightarrow{\lambda} (q', \beta') \in \mathcal{R}$ (i) $q = q'$, (ii) $\beta = A \in \mathcal{A}$, (iii) if $\lambda = \nu(q'', \beta'')$ then $q'' = q$ and $\beta'' \in \mathcal{A}$, and (iv) $\beta' = \epsilon$ unless $\lambda = \epsilon$, in which case: (v) $\beta' \in \{\epsilon, BC : B, C \in \mathcal{A}\}$.

Proposition 1. Given an ACPS \mathcal{P} , a simple coverability query Q and a $\Pi^0 \triangleleft \Gamma^0$ there exist ACPS $\mathcal{F}(\mathcal{P})$ in normal form, a simple coverability query $\mathcal{F}(Q)$, and $\mathcal{F}(\Pi^0 \triangleleft \Gamma^0)$ — all polynomial-time computable — such that: Q is a yes-instance if, and only if, $\mathcal{F}(Q)$ is a yes-instance; and \mathcal{P} is bounded (terminating) from $\Pi^0 \triangleleft \Gamma^0$ if, and only if, $\mathcal{F}(\mathcal{P})$ is bounded (terminating respectively) from $\mathcal{F}(\Pi^0 \triangleleft \Gamma^0)$.

Henceforth we shall elide the single state q and write a rule simply as $\beta \xrightarrow{\lambda} \beta'$. There is a well-known connection between pushdown systems and *context-free grammars* (CFG). It is trivial to see that transitions of a single ACPS process in normal form are (essentially) left-most derivations of a CFG.

Definition 3. Let Σ be an alphabet of terminal symbols. A *context-free grammar* (CFG) is a triple $\mathcal{G} = (\Sigma, \mathcal{N}, \mathcal{R})$ where \mathcal{R} comprises rewrite rules of the form: (i) $X \rightarrow YZ$ where $X, Y, Z \in \mathcal{N}$ and (ii) $X \rightarrow a$ where $a \in \Sigma \cup \{\epsilon\}$.

Fix an ACPS $\mathcal{P} = (\{q\}, \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R})$ in normal form. From \mathcal{P} we obtain a CFG $\mathcal{G}(\mathcal{P}) = (\Sigma(\mathcal{P}), \mathcal{N}(\mathcal{P}), \mathcal{R}(\mathcal{P}))$ whose non-terminals $\mathcal{N}(\mathcal{P})$ are the stack alphabet \mathcal{A} , the set of terminals $\Sigma(\mathcal{P})$ is trivially obtained from Λ (we only replace $\nu(q, A)$ by νA), and its rules $\mathcal{R}(\mathcal{P})$ are derived by

$$\begin{aligned} A \xrightarrow{\nu A'} \epsilon &\mapsto A \rightarrow \nu A' & A \xrightarrow{c!m} \epsilon &\mapsto A \rightarrow c!m \\ A \xrightarrow{\epsilon} \epsilon &\mapsto A \rightarrow \epsilon & A \xrightarrow{c?m} \epsilon &\mapsto A \rightarrow c?m \\ A \xrightarrow{\epsilon} BC &\mapsto A \rightarrow BC \end{aligned}$$

In moving from the operational view of pushdown systems to CFGs it is possible to build upon known results on how CFGs interact with reorderings in the words they produce. In our setting, channels are unordered and a new process may start at its own leisure. Hence, the execution order of concurrency side-effects such as send and spawn is immaterial. However, the sequencing of other side-effects, notably message retrieval which requires synchronisation, is observable.

Exploiting Commutativity: The use of an *independence relation* is a common technique to formalise such sensitivity to order. An independence relation I over a set U is a symmetric irreflexive relation over U . It induces a congruence relation \simeq_I on U^* defined as the least equivalence relation \mathcal{R} such that: $(\beta, \beta') \in \mathcal{R} \implies \forall \gamma, \gamma' \in U^* : (\gamma \beta \gamma', \gamma \beta' \gamma) \in \mathcal{R}$ and $I \subseteq \mathcal{R}$. An element $u \in U$ is said to be *non-commutative wrt to I* if $(u, u') \notin I$ for all $u' \in U$ and we denote the set of non-commutative elements by $U^{\text{non-com}}$. Similarly we call a $u \in U$ *commutative wrt to I* if $(u, u') \in I$ for all $u' \in U \setminus U^{\text{non-com}}$, i.e. commutative elements commute with all elements except non-commutative ones, and write U^{com} for the set of commutative elements. An independence relation I that partitions U into commutative and non-commutative elements is said to be *unambiguous*. Let $\Xi \subseteq U$, then we define the independence relation generated by Ξ as

$$\text{IndRel}_U(\Xi) := \{(a, a'), (a', a) \mid a, a' \in \Xi, a \neq a'\}.$$

Since receive is the only blocking concurrency action, we specify the actions that we want to commute as $\Sigma^{\text{com}} := \Sigma(\mathcal{P}) \setminus \{c?m \mid c \in \text{Chan}, m \in \text{Msg}\}$. Then the independence relation $\text{IndRel}_{\Sigma(\mathcal{P})}(\Sigma^{\text{com}})$ allows us to commute all concurrency actions *except* receive. Further, $\text{IndRel}_{\Sigma(\mathcal{P})}(\Sigma^{\text{com}})$ is unambiguous and partitions $\Sigma(\mathcal{P})$ into the commutative and non-commutative actions Σ^{com} and $\Sigma^{\text{non-com}}$ respectively.

We expect an independence relation over non-terminals, that is consistent with $\text{IndRel}_{\Sigma(\mathcal{P})}(\Sigma^{\text{com}})$, to classify a non-terminal as commutative if it is productive and rewrites only to commutative symbols. This intuition can be captured in terms of a suitable monotone function $F : \mathcal{P}[\mathcal{N}(\mathcal{P})] \rightarrow \mathcal{P}[\mathcal{N}(\mathcal{P})]$. For a subset $U \subseteq \mathcal{N}(\mathcal{P})$, a non-terminal N is an element of the set $F(U)$ just if (i) $\mathcal{L}(N) \neq \emptyset$, and (ii) for each w such that $N \rightarrow w$, $\Sigma(w) \subseteq \Sigma^{\text{com}}$ and $\mathcal{N}(w) \subseteq U$, writing $\Sigma(w)$ (respectively $\mathcal{N}(w)$) for the set of terminal (respectively non-terminal) symbols that occur in the word w . We define \mathcal{N}^{com} as the greatest fixpoint of F . Similarly to the case of concurrency actions, this choice of commutative non-terminals

gives rise to the independence relation $I(\mathcal{G}(\mathcal{P}))$ defined by

$$I(\mathcal{G}(\mathcal{P})) := \text{IndRel}_{\mathcal{N}(\mathcal{P}) \cup \Sigma(\mathcal{P})}(\Sigma^{\text{com}} \cup \mathcal{N}^{\text{com}}).$$

Again, the independence relation $I(\mathcal{G}(\mathcal{P}))$ yields a partition of $\mathcal{N}(\mathcal{P}) \cup \Sigma(\mathcal{P})$ into the commutative $(\Sigma^{\text{com}} \cup \mathcal{N}^{\text{com}})$ and non-commutative symbols $\Sigma^{\text{non-com}} \cup \mathcal{N}^{\text{non-com}}$. The set $\mathcal{N}^{\text{non-com}}$ denotes the set of non-commutative non-terminals and it is easy to see that $\mathcal{N}^{\text{non-com}} = \mathcal{N}(\mathcal{P}) \setminus \mathcal{N}^{\text{com}}$. Commutative non-terminals are productive and only produce commutative symbols. Rewriting non-commutative non-terminals may block. For example, a non-terminal is an element of \mathcal{N}^{com} if it produces only finite words of sends and spawns (possibly infinitely many), or infinite words of sends and spawns including all their prefixes. By contrast, a non-terminal that produces a receive terminal or no finite word is categorised as non-commutative.

A CFG $(\Sigma, \mathcal{N}, \mathcal{R})$ endowed with an independence relation on $\mathcal{N} \cup \Sigma$ is called a *partially commutative context-free grammar* (PCCFG) [9]. In previous work, we introduced a class of PCCFG, *asynchronous partially commutative pushdown systems* (APCPS), defined over $\Sigma(\mathcal{P})$ and endowed with $I(\mathcal{G})$ [29]. Even though APCPS originate from a grammar they induce a transition system semantics that is suitable for the analysis of asynchronous concurrent pushdown systems.

Definition 4. An *asynchronous partially commutative pushdown system* (APCPS) is a PCCFG $\mathcal{G} = (\Sigma(\mathcal{P}), I(\mathcal{G}), \mathcal{N}, \mathcal{R})$.

In particular endowing $\mathcal{G}(\mathcal{P})$ with $I(\mathcal{G}(\mathcal{P}))$ yields an APCPS. Given an APCPS \mathcal{G} the *standard semantics* gives rise to a transition system very similar to an ACPS. The main difference is that processes are equivalence classes induced by $\simeq_{I(\mathcal{G})}$.

Standard Semantics: In the standard semantics a process, denoted γ , is a word $\delta \beta X_1 \beta_1 \cdots X_n \beta_n$ (modulo $\simeq_{I(\mathcal{G})}$) where $X_i \in \mathcal{N}^{\text{non-com}}$ and $\beta, \beta_i \in \mathcal{N}^{\text{com}}$ and $\delta \in \Sigma \cup \mathcal{N} \cup \{\epsilon\}$. We call the set of such processes *Procs*. The standard semantics of an APCPS is a transition system over $\mathbb{M}[\text{Procs}] \times \text{Chans}$ where $\text{Chans} = (\text{Chan} \rightarrow \mathbb{M}[\text{Msg}])$. We order processes $\delta \pi_0 \leq_{\text{Procs}} \delta \pi_1$ if $\delta \pi_0' \leq_{(\mathcal{N} \cup \Sigma)^*} \delta \pi_1'$ for some π_0' and π_1' such that both $\delta \pi_i \simeq_{I(\mathcal{G})} \delta \pi_i'$. We lift \leq_{Procs} to a preorder \leq_{APCPS} on configurations, using the multiset and function extension, and obtain a PSTS $(\mathbb{M}[\text{Procs}] \times \text{Chans}, \rightarrow_{\text{con}}, \leq_{\text{APCPS}})$ where the transition relation \rightarrow_{con} is defined in the left column of Table I. Processes change state by performing a leftmost CFG derivation of \mathcal{G} until an action appears in the leftmost position (Rules (R-1) and (R-2)); the type of action then determines a concurrency side-effect that interacts with the rest of the configuration (Rules (R-3)–(R-5)) causing the action to be consumed and enabling further leftmost reductions of \mathcal{G} . We say an APCPS coverability query $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ is simple if $\pi \simeq_{I(\mathcal{G})} A \in \mathcal{N}$ for all π in Π and Π_0 .

The Shaped Constraint. If $\beta_i \in \mathcal{N}^{\text{com}*}$, $X_i \in \mathcal{N}^{\text{non-com}}$, $\delta \in \mathcal{N}$, $\bar{\delta} \in \mathcal{N} \cup \Sigma \cup \{\epsilon\}$, and $n \leq K$, we say an ACPS process $\pi = \delta \beta_0 X_1 \beta_1 \cdots X_n \beta_n$ (an APCPS process $\bar{\pi} \simeq_{I(\mathcal{G})} \bar{\delta} \beta X_1 \beta_1 \cdots X_n \beta_n$) is *K-shaped*. An ACPS (APCPS) configuration $\Pi \triangleleft \Gamma$ is *K-shaped* if all processes in Π are *K-shaped* and we say an ACPS \mathcal{P} (an APCPS \mathcal{G}) has *K-shaped stacks from $\Pi_0 \triangleleft \Gamma_0$* just if all reachable configurations from

Standard semantics	Alternative semantics
(R-1): $A \gamma \parallel \Pi \triangleleft \Gamma \rightarrow_{\text{con}} \beta \gamma \parallel \Pi \triangleleft \Gamma$	(R'-1): $A M \gamma' \parallel \Pi' \triangleleft \Gamma \rightarrow_{\text{con}'} \beta M \gamma' \parallel \Pi' \triangleleft \Gamma$
(R-2): $A \gamma \parallel \Pi \triangleleft \Gamma \rightarrow_{\text{con}} B C \gamma \parallel \Pi \triangleleft \Gamma$	(R'-2): $A M \gamma' \parallel \Pi' \triangleleft \Gamma \rightarrow_{\text{con}'} B (\mathbb{M}(w) \oplus M) \gamma' \parallel \Pi' \triangleleft \Gamma$
(R-3): $(c ? m) \gamma \parallel \Pi \triangleleft ([m] \oplus q)^c, \Gamma \rightarrow_{\text{con}} \gamma \parallel \Pi \triangleleft q^c, \Gamma$	(R'-3): $(c ? m) \gamma' \parallel \Pi' \triangleleft ([m] \oplus q)^c, \Gamma \rightarrow_{\text{con}'} \gamma' \parallel \Pi' \triangleleft q^c, \Gamma$
(R-4): $(c ! m) \gamma \parallel \Pi \triangleleft q^c, \Gamma \rightarrow_{\text{con}} \gamma \parallel \Pi \triangleleft ([m] \oplus q)^c, \Gamma$	(R'-4): $(c ! m) \gamma' \parallel \Pi' \triangleleft q^c, \Gamma \rightarrow_{\text{con}'} \gamma' \parallel \Pi' \triangleleft ([m] \oplus q)^c, \Gamma$
(R-5): $(\nu X) \gamma \parallel \Pi \triangleleft \Gamma \rightarrow_{\text{con}} \gamma \parallel X \parallel \Pi \triangleleft \Gamma$	(R'-5): $(\nu X) \gamma' \parallel \Pi' \triangleleft \Gamma \rightarrow_{\text{con}'} \gamma' \parallel X \parallel \Pi' \triangleleft \Gamma$
	(R'-6): $M X \gamma' \parallel \Pi' \triangleleft \Gamma \rightarrow_{\text{con}'} X \gamma' \parallel \Pi' \parallel \Pi(M) \triangleleft \Gamma \oplus \Gamma(M)$
	(R'-7): $M' \gamma' \parallel \Pi' \triangleleft \Gamma \rightarrow_{\text{con}'} M'' \gamma' \parallel \Pi' \parallel \Pi(M') \triangleleft \Gamma \oplus \Gamma(M')$

Let (\dagger) be a condition on a $\beta \in \mathcal{N}^*$: $\beta = BC$ and C is commutative. Rules (R-2) and (R'-2) have a side condition: $A \rightarrow BC \in \mathcal{G}$, BC satisfies (\dagger) , and $C \rightarrow^* w$. Rules (R-1) and (R'-1) have a side condition: $A \rightarrow \beta \in \mathcal{G}$ and β does *not* satisfy (\dagger) . In rule (R'-6) we require the multiset $M \in \mathbb{M}[\Sigma^{\text{com}}]$ whereas in rule (R'-7) M' may be an element of $\mathbb{M}[\Sigma^{\text{com}} \cup \mathcal{N}]$ and $M'' = M' \upharpoonright \mathcal{N}$. We use the abbreviations: $\Pi(M) := \parallel_{A \in \mathcal{N}} \parallel_1^{M(\nu A)} A$, and $\Gamma(M) := \bigoplus_{c \in \text{Chan}} \bigoplus_{m \in \text{Msg}} [c \mapsto [m^{M(c!m)}]]$.

Table I: Transition semantics for APCPS

$\Pi_0 \triangleleft \Gamma_0$ are K -shaped. Intuitively, the shaped constraint requires that, at all times, at most an *a priori* fixed number K of non-commutative non-terminals K may reside in the stack. Because the restriction does not apply to commutative non-terminals, stacks can grow to arbitrary heights.

Given a simple coverability query $Q = (\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ for an ACPS in normal form \mathcal{P} we may analyse $Q' = (\mathcal{G}(\mathcal{P}), \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ a simple coverability query for the APCPS $\mathcal{G}(\mathcal{P})$ instead; the reduction preserves the shape constraint:

Proposition 2. Q is a yes-instance, if and only if, Q' is a yes-instance. Hence simple coverability for ACPS and APCPS polynomial-time inter-reduce. Further \mathcal{P} is K -shaped from $\Pi_0 \triangleleft \Gamma_0$ if, and only if, $\mathcal{G}(\mathcal{P})$ is K -shaped from $\Pi_0 \triangleleft \Gamma_0$.

Deciding coverability, boundedness or termination on the standard semantics looks daunting. Even a K -shaped process is infinite state, follows a stack discipline and may synchronise with an unbounded number of other processes. Since APCPS subsume ordinary concurrent pushdown systems, simple coverability is undecidable in general for the standard semantics. However, the independence relation $\simeq_{I(\mathcal{G})}$ enables a simplification which is formalised in the *alternative semantics*. The key idea is to summarise the effects of commutative non-terminals. In the alternative semantics, rather than keeping track of the contents of the stack, we precompute the actions produced by commutative non-terminals and store them in summaries on the stack. The non-commutative procedure calls, which are left on the stack, then act as separators for these summaries.

Alternative Semantics: In the alternative semantics a process, which we denote by γ' , has the shape $\delta M X_1 M_1 \cdots X_n M_n$, with $X_i \in \mathcal{N}^{\text{com}}$, $M, M_i \in \mathbb{M}[\mathcal{N} \cup \Sigma^{\text{com}}]$ and $\delta \in \Sigma \cup \mathcal{N} \cup \{\epsilon\}$, and is said to be K -shaped if $n \leq K$. We denote the set of such processes by Procs' . Then the alternative concurrent semantics of an APCPS is a transition system over elements of $\mathbb{M}[\text{Procs}'] \times \text{Chans}$. We abbreviate a set of alternative processes running in parallel as Π' .

The right column of Table I shows the alternative semantics along-side the standard semantics. Most transition rules, barring the different shape of processes, are essentially the same as the standard semantics. The rules that are different implement and manage the summary of commutative non-terminals.

Rule (R'-2) executes a rule $A \rightarrow BC$ by precomputing the actions w of the commutative non-terminal C and inserting w 's *Parikh image* $\mathbb{M}(w)$ into the summary M . This is the counterpart of a push in the alternative semantics. The rules (R'-6) and (R'-7) are the pop counterparts; they ensure that the precomputed actions are rendered effective at the appropriate moment. Rule (R'-6) is applicable when the summary M contains exclusively commutative actions; such a summary denotes a sequence of commutative non-terminals whose computation terminates and generates concurrency actions. Rule (R'-7) handles the case where the summary M' contains non-terminals. Such a summary represents a partial computation of a sequence of commutative non-terminals. In this case rule (R'-7) dispatches all commutative actions and then blocks. It is necessary to consider this case since not all non-terminals have terminating computations. Thus rule (R'-2) may non-deterministically abandon the pre-computation of actions.

Again we can turn \mathcal{G} with $\rightarrow_{\text{con}'}$ into a PSTS $(\mathbb{M}[\text{Procs}'] \times \text{Chans}, \rightarrow_{\text{con}'}, \leq_{\text{APCPS}'})$ by endowing it with a preorder $\leq_{\text{APCPS}'}$. We order elements of $\mathbb{M}[\Sigma \cup \mathcal{N}]$ with the usual multi-set ordering $\leq_{\mathbb{M}[\Sigma \cup \mathcal{N}]}$ and elements of Procs' are ordered by $\leq_{\text{Procs}'} := (\Sigma \cup \mathcal{N} \cup \mathbb{M}[\Sigma \cup \mathcal{N}])^*$ which is lifted to configurations as before. We say a coverability query $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, \Pi' \triangleleft \Gamma)$ for the alternative semantics is *simple* if $\pi' = A \in \mathcal{N}$ for all $\pi' \in \Pi'$ and Π_0 . The standard and alternative semantics give rise to the same yes-instances of simple coverability:

Theorem 1 ([29]). *A query Q is a yes-instance for simple coverability on the standard semantics if, and only if, Q is a yes-instance on the alternative semantics.*

The alternative semantics for shape-constrained APCPS gives rise to a *well-structured transition system* (WSTS) [20], which implies the decidability of simple coverability for both alternative and standard semantics [29]. However, this result yields only coarse complexity results. To study the complexity of simple coverability, we introduce a non-trivial extension of Petri nets, *nets with nested coloured tokens*.

III. NETS WITH NESTED COLOURED TOKENS

The alternative operational semantics for APCPS requires the ability to model configurations that contain multisets of

multisets — a capability that appears to be beyond Petri nets. Rules (R'-6) and (R'-7) seem to require a feature that allows the transfer or “ejection” of elements from inside a nested multiset. Fortunately, we know from the literature [16, 22, 19] that computing the summary of a commutative non-terminal, as performed by rule (R'-2), can be achieved by a Petri net.

Inspired by *nested Petri nets* [35], which give rise to configurations of nested structures of multisets, we introduce *nets with nested coloured tokens* (NNCT) which feature multisets of multisets and vertical transfers. NNCT is designed with the necessary features to implement the alternative semantics for APCPS, while also allowing APCPS to simulate NNCT.

Definition 5. A net with nested coloured tokens (NNCT) is a quintuple $\mathcal{N} = (\mathcal{P}^S, \mathcal{P}^C, \mathcal{P}^{col}, \mathcal{R}, \zeta)$ where $\mathcal{P}^S, \mathcal{P}^C, \mathcal{P}^{col}$ and \mathcal{R} are the finite sets of *simple places*, *complex places*, *colours*, and *rules*; and the *colour mapping* ζ is a map from \mathcal{P}^{col} to \mathcal{P}^S .

Markings. We define *markings* for $\mathcal{P}^S, \mathcal{P}^{col}$ and \mathcal{P}^C as

$$\begin{aligned}\mathcal{M}^{simple} &:= \{m \mid m : \mathcal{P}^S \rightarrow \mathbb{M}[\{\bullet\}]\} \\ \mathcal{M}^{col} &:= \{m \mid m : \mathcal{P}^{col} \rightarrow \mathbb{M}[\{\bullet\}]\} \\ \mathcal{M}^{complex} &:= \{m \mid m : \mathcal{P}^C \rightarrow \mathbb{M}[\mathcal{M}^{col}]\}.\end{aligned}$$

We also call a marking for \mathcal{P}^{col} (i.e. an element of \mathcal{M}^{col}) a *complex token*, which we view as a multiset of *coloured tokens*. Thus a marking for \mathcal{P}^C fills each complex place with a multiset of complex tokens. A *configuration* of a NNCT is the disjoint union of a marking for \mathcal{P}^S and a marking for \mathcal{P}^C , i.e.

$$Config := \{m + m' \mid m \in \mathcal{M}^{simple}, m' \in \mathcal{M}^{complex}\}.$$

Rules. We partition \mathcal{R} into *SimpleRules*, *ComplexRules* and *TransferRules*. A rule $r \in SimpleRules$ is a pair (I, O) such that $I, O \in Config$ where $I(p)(m) = 0$ if both $p \in \mathcal{P}^C$ and $m \neq \mathbf{0}$, where, as a reminder, $\mathbf{0}(x) = \emptyset$ for any x . A rule $r \in ComplexRules$ is a pair $((p, I), (p', c, O))$ such that $I, O \in Config_S := \{m + \mathbf{0} \mid m \in \mathcal{M}^{simple}\}$; $p, p' \in \mathcal{P}^C$ and $c \in \mathcal{M}^{col}$. A rule $r \in TransferRules$ is a pair $((p, I), (p', P, O))$ such that $p, p' \in \mathcal{P}^C$; $I, O \in Config_S$ and $P \subseteq \mathcal{P}^{col}$ such that $\zeta \upharpoonright P : P \rightarrow \zeta(P)$ is bijective, i.e., ζ^{-1} is well-defined on $\zeta(P)$. We sometimes need to refer to the mappings I, O of a rule r , for which we write I_r and O_r .

Operational Semantics. Let $s \in Config$.

(R1) Suppose $r = (I, O) \in SimpleRules$. If rule r is *enabled* at s i.e. $s = s_0 \oplus I$ for some $s_0 \in Config$, then $s \xrightarrow{r}_{\mathcal{N}} s'$ where $s' := s_0 \oplus O$.

(R2) Suppose $r = ((p, I), (p', c, O)) \in ComplexRules$. If rule r is *enabled* at s i.e. $s = s_0 \oplus I \oplus [p \mapsto [m]]$ for some $s_0 \in Config$, then $s \xrightarrow{r}_{\mathcal{N}} s'$ where $s' = s_0 \oplus O \oplus [p' \mapsto [m \oplus c]]$.

(R3) Suppose $r = ((p, I), (p', P, O)) \in TransferRules$. If rule r is *enabled* at s i.e. $s = s_0 \oplus I \oplus [p \mapsto [m]]$ for some $s_0 \in Config$, then $s \xrightarrow{r}_{\mathcal{N}} s'$ such that

$$s' = s_0 \oplus O \oplus (m_P \circ \zeta^{-1}) \oplus [p' \mapsto [m_{\bar{P}}]]$$

where $m_P = m \upharpoonright P$ and $m_{\bar{P}} = m \upharpoonright (\mathcal{P}^{col} \setminus P)$.

A simple rule may insert new complex tokens into complex places; it may also remove empty complex tokens. A complex

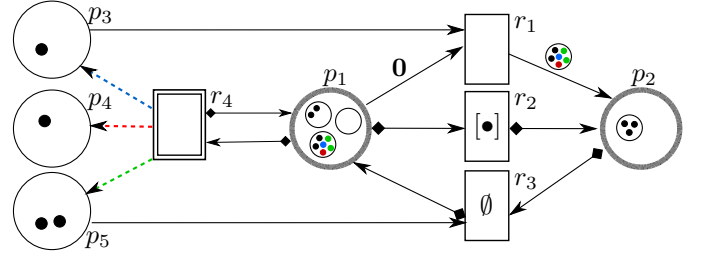


Figure 3: The net of Example 1

rule removes a complex token m from a complex place p and inserts a new complex token $m \oplus c$ to a complex place p' . A set of *active colours* $P \subseteq \mathcal{P}^{col}$ is associated to each transfer rule, which removes a complex token m from a complex place p , and inserts into p' the complex token $m_{\bar{P}}$ which is obtained from m less all tokens with an active colour c ; for each such $c \in P$, these tokens are transferred to the simple place $\zeta(c)$ which corresponds to a *multiset-addition* of $m_P \circ \zeta^{-1}$.

Subclasses of NNCT. Let $\mathcal{N} = (\mathcal{P}^S, \mathcal{P}^C, \mathcal{P}^{col}, \mathcal{R}, \zeta)$ be an NNCT. If a transfer rule $r = ((p, I), (p', P, O))$ is such that $P = \mathcal{P}^{col}$ then we say r is a *total transfer rule*. If all transfer rules of \mathcal{N} are total then we say \mathcal{N} is a *total transfer NNCT*.

Example 1. We define a NNCT \mathcal{N} with places: $\mathcal{P}^C = \{p_1, p_2\}$, $\mathcal{P}^S = \{p_3, p_4, p_5\}$, and $\mathcal{P}^{col} = \{red, green, blue, black\}$, and colour mapping: $\zeta : red \mapsto p_4, green \mapsto p_5, blue \mapsto p_3$. The rule $r_1 = (\{p_1 \mapsto [\mathbf{0}], p_3 \mapsto [\bullet]\}, \{p_2 \mapsto [m_1]\})$ is simple where the complex token $m_1 = \{black \mapsto [\bullet^2], blue \mapsto [\bullet], green \mapsto [\bullet^2], red \mapsto [\bullet]\}$. The complex rules are:

$$\begin{aligned}r_2 &= ((p_1, \emptyset), (p_2, \{black \mapsto [\bullet]\}, \emptyset)) \\ r_3 &= ((p_2, \{p_5 \mapsto [\bullet]\}), (p_1, \emptyset, \emptyset))\end{aligned}$$

The transfer rule is $r_4 = ((p_1, \emptyset), (p_1, \{red, blue, green\}, \emptyset))$. Note that \mathcal{N} is *not* a total transfer NNCT. We graphically represent NNCT similarly to Petri nets. In Figure 3 we show a configuration of \mathcal{N} . The complex place p_1 contains the complex token m_1 , the empty complex token $\mathbf{0}$ (displayed as an empty circle) and the complex token $m_2 = \{black \mapsto [\bullet^2]\}$. The complex token $m_3 = \{black \mapsto [\bullet^3]\}$ is located in complex place p_2 . We distinguish transfer rules by using double edged boxes and we display ζ and the set of active colours using dashed arrows. We indicate the origin and destination for complex tokens moved by complex and transfer rules with additional arcs that have a \blacklozenge end. Boxes for complex rules are labelled with the colour marking c to inject. Simple rules may have arcs *from* complex places labelled with $\mathbf{0}$ (indicating the removal of $\mathbf{0}$) and arcs *to* complex places labelled with the complex token to be added. Rule r_1 removes an empty complex token from p_1 and a simple token from p_3 , and adds the complex token m_1 to p_2 . The complex rule r_2 non-deterministically selects a complex token e.g. m_2 and moves it to p_2 while inserting one *black*-coloured token, i.e. m_2 becomes m_3 . Let us imagine for a moment that the simple places p_3, p_4 , and p_5 are all empty. Rule r_4 non-deterministically selects a complex token in p_1 , m_1 say, and distributes its *blue*, *green* and *red*-coloured tokens as indicated

by the coloured dashed arrows to the simple places p_3, p_4 and p_5 and turns them into simple (black) tokens. The result is that p_3 and p_4 contain a simple token each and p_5 two as displayed in Figure 3; the token m_1 , less its *blue, green* and *red*-coloured tokens, remains in place p_1 and becomes m_2 .

NNCT are WSTS. We recall a few definitions: let (U, \leq) be a preordered set; we say \leq is a *well-quasi-order* (WQO) if for all infinite sequences u_1, u_2, \dots there exist $i < j$ such that $u_i \leq u_j$. A WSTS is a PSTS $\mathcal{S} = (S, \rightarrow_{\mathcal{S}}, \leq_{\mathcal{S}})$ such that $\leq_{\mathcal{S}}$ is a WQO and $\rightarrow_{\mathcal{S}}$ is monotone with respect to $\leq_{\mathcal{S}}$; a WSTS is said to be *strict* if $\rightarrow_{\mathcal{S}}$ is strictly monotone: we say $\rightarrow_{\mathcal{S}}$ is (strictly) *monotone* if $s \rightarrow_{\mathcal{S}} s'$ and $s <_{\mathcal{S}} t$ implies that there exists t' such that $t \rightarrow_{\mathcal{S}} t'$ and $s' \leq_{\mathcal{S}} t'$ ($s' <_{\mathcal{S}} t'$ respectively). We equip Config and \mathcal{M}^{col} with preorders: let $m, m' \in \mathcal{M}^{\text{col}}$, we define $m \leq_{\mathcal{M}^{\text{col}}} m'$ if for all $p^{\text{col}} \in \mathcal{P}^{\text{col}}$ we have either $0 = |m(p^{\text{col}})| = |m'(p^{\text{col}})|$, or $0 < |m(p^{\text{col}})| \leq |m'(p^{\text{col}})|$. For $s, s' \in \text{Config}$ we define $s \leq_{\text{Config}} s'$ if for all $p \in \mathcal{P}^S$ we have $s(p) \leq \mathbb{M}[\{\bullet\}] s'(p)$ and $\forall p' \in \mathcal{P}^C$ we have $s(p') \leq \mathbb{M}[\mathcal{M}^{\text{col}}] s'(p')$.

Lemma 2. $(\mathcal{M}^{\text{col}}, \leq_{\mathcal{M}^{\text{col}}})$ and $(\text{Config}, \leq_{\text{Config}})$ are WQO.

Thanks to the use of the refined order $\leq_{\mathcal{M}^{\text{col}}}$ the transition relation $\rightarrow_{\mathcal{N}}$ for a NNCT \mathcal{N} is strictly monotone:

Proposition 3. $(\text{Config}, \rightarrow_{\mathcal{N}}, \leq_{\text{Config}})$ is a strict WSTS.

Let $\uparrow S = \{s : \exists s_0 \in S, s_0 \leq_{\mathcal{S}} s\}$ and $\text{Pred}(S) = \{s : s \rightarrow_{\mathcal{S}} s', s' \in S\}$. For a WSTS \mathcal{S} , coverability, termination and boundedness are decidable [20] provided that $\uparrow \text{Pred}(\uparrow \{s\})$ is effectively computable for any $s \in S$, the WQO $\leq_{\mathcal{S}}$ is decidable and (for boundedness:) \mathcal{S} is a strict WSTS.

Theorem 2. *Coverability, termination and boundedness are decidable for NNCT.*

By construction, NNCT can implement the alternative semantics of APCPS and *vice versa*. We encode processes as complex tokens and allocate a colour for each pair in $\Sigma \times \{0, \dots, K\}$ in order to encode summaries as coloured tokens. Channels are encoded in simple places and rule (R'-2) is implemented by a slightly modified CCFG widget *à la* Ganty and Majumdar [22]. The resulting NNCT has $O(n \cdot |\Sigma| |\mathcal{N}|)$ simple places, $O((|\mathcal{N}| |\Sigma|)^{O(K)})$ complex places, $O(K \cdot |\Sigma|)$ colours and $O(n \cdot |\mathcal{R}| \cdot (|\mathcal{N}| |\Sigma|)^{O(K)})$ rules where $A_1 \parallel \dots \parallel A_n \triangleleft \Gamma$ is the configuration to be covered.

Theorem 3. *Simple coverability for K -shaped APCPS in the alternative semantics EXPTIME-time reduces to NNCT coverability.*

Given a total-transfer NNCT \mathcal{N} , we can construct a simulating APCPS \mathcal{G} that uses channels to encode simple places. A process with a single summary simulates a complex token and the coloured tokens it carries. \mathcal{N} runs a control process that executes rules by manipulating the state of processes and channels by communication through auxiliary channels. An injection of a multiset of coloured tokens c by a complex rule is simulated by forcing a process π to add the summary of a non-terminal representing c to π 's summary. A transfer rule is simulated by forcing a process encoding a complex token to dispatch its

summary. Simple coverability may then be decided on \mathcal{G} . A NNCT coverability query $Q = (\mathcal{N}, s_0, s_{\text{cov}})$ is *simple*, if s_0 and s_{cov} contain *no* complex tokens: $\forall p \in \mathcal{P}^C \ s_0(p) = s_{\text{cov}}(p) = 0$.

Theorem 4. *Simple coverability, boundedness and termination for a total-transfer NNCT EXPTIME reduces to simple coverability, boundedness and termination respectively for a 4-shaped APCPS in the alternative semantics.*

We show in the next two sections that NNCT coverability is TOWER-complete. TOWER was recently introduced by Schmitz to provide a meaningful complexity class for decision problems of non-elementary complexity [40]. The notion of reduction used in TOWER are ELEMENTARY-time reductions which give rise to TOWER-complete problems. The EXPTIME reductions of Theorem 3 and Theorem 4 are thus sufficient to infer that coverability for APCPS is TOWER-complete.

IV. UPPER BOUND: A NESTED RACKOFF ARGUMENT.

Let us fix a NNCT $\mathcal{N} = (\mathcal{P}^S, \mathcal{P}^C, \mathcal{P}^{\text{col}}, \mathcal{R}, \zeta)$ with a coverability query $(\mathcal{N}, s_0, s_{\text{cov}})$ throughout this section. We enumerate \mathcal{N} 's simple places $\mathcal{P}^S = \{p_{(1)}, \dots, p_{(n_s)}\}$, complex places $\mathcal{P}^C = \{p'_{(1)}, \dots, p'_{(n_c)}\}$ and colours $\mathcal{P}^{\text{col}} = \{p^{\text{col}}_{(1)}, \dots, p^{\text{col}}_{(n_{\text{col}})}\}$.

Our proof of TOWER-membership for NNCT coverability is inspired by Rackoff's method [37]. The Rackoff method constructs a bound on the *covering radius* of a given target state s_{cov} , which is the greatest *covering distance* to s_{cov} from an arbitrary state s , where the covering distance is the length of the shortest covering path. Typically the covering radius can then be used to establish a bound \mathcal{B} on the space required for a machine representation of any state along a covering path for s_{cov} . Depending on the flavour of VAS, it is then easy to see that a \mathcal{B} -space bounded non-deterministic/alternating Turing machine can find a covering path, if there is one, and reject a path if its length exceeds the covering radius.

Definition 6. Suppose $\mathcal{S} = (S, \rightarrow_{\mathcal{S}}, \leq_{\mathcal{S}})$ is a PSTS. We say $s \in S^*$ is a *path* if for all $1 \leq i < |s|$, $s(i) \rightarrow_{\mathcal{S}} s(i+1)$. A path s is *covering* for s' from s if $s' \leq_{\mathcal{S}} s(|s|)$ and $s(1) = s$.

We define $\text{dist}_{\mathcal{S}}(s, s')$, the *covering distance* between s and s' , as follows: if there exists a covering path for s' from s then

$$\text{dist}_{\mathcal{S}}(s, s') := \min \{|s| \mid s \text{ covering path for } s' \text{ from } s\};$$

otherwise set $\text{dist}_{\mathcal{S}}(s, s') := 0$. We define the *covering radius* for s' as $\rho_{\mathcal{S}}(s') := \sup \{\text{dist}_{\mathcal{S}}(s, s') \mid s \in S_{\mathcal{S}}\}$, and the *covering diameter* of a subset $S' \subseteq S_{\mathcal{S}} \times S_{\mathcal{S}}$, $d_{\mathcal{S}}(S') := \sup \{\text{dist}_{\mathcal{S}}(s, s') \mid (s, s') \in S'\}$, i.e. the maximum covering distance between any pair in S' .

To determine a bound on the covering radius a more general, relativised problem is considered: suppose the contents of the last $n - i$ places are *ignored* and the first i places are *not ignored*, how can we bound the covering radius ρ_i for s_{cov} ?

One way to ignore the contents of simple places in \mathcal{N} is to let their contents be “unbounded”: define $\mathcal{M}^{\text{simple}\infty} = \mathcal{P}^S \rightarrow \mathbb{M}^{\infty}[\{\bullet\}]$ where we write $\mathbb{M}^{\infty}[U]$ for multisets over elements in U with possibly infinite multiplicity, i.e. the set of functions $U \rightarrow \mathbb{N}^{\infty}$. Configurations with possibly infinite

simple markings can be defined as $Config^\infty = \{m+m' \mid m \in \mathcal{M}^{simple^\infty}, m' \in \mathcal{M}^{complex}\}$. We extend the transition relation $\rightarrow_{\mathcal{N}}$ in the obvious way to $Config^\infty$. For each $i \leq n_s$ we then define a new transition relation $\rightarrow_{\mathcal{N}_i}$ that formalises that we ignore the last $n_s - i$ simple places i.e. $s \rightarrow_{\mathcal{N}_i} s'$ just if $s \rightarrow_{\mathcal{N}} s'$ and for all $i < j \leq n_s$ $s(p_{(j)})(\bullet) = s'(p_{(j)})(\bullet) = \infty$. Writing $\mathcal{N}_i = (Config^\infty, \rightarrow_{\mathcal{N}_i}, \leq_{Config})$ the relativised problem is then to determine the covering radius $\rho_{\mathcal{N}_i}(s_{cov})$.

The core argument underlying the general Rackoff method establishes a recurrence relation that relates ρ_{i+1} to ρ_i . This is achieved by a careful analysis of the size or *norm* of configurations that occur along covering paths for s_{cov} . In general, we say $\|-\|$ is a *norm* for a set S if $\|-\|$ is a function from S to \mathbb{N}^∞ and we extend $\|-\|$ to a norm $\|-\|^*$ on sequences S^* in the usual way $\|s\|^* = \max\{\|s\| : 1 \leq i \leq |s|\}$. The general Rackoff method introduces a family of norms $(\|-\|_i)_{i=0}^n$ where each $\|-\|_i$ ignores the contents of the last $(n-i)$ places. Two facts are then established: (i) there exists an upper bound $B(\rho_i)$ on how many tokens can be removed in a path of length ρ_i ; and (ii) for any covering path s for s_{cov} one of two cases apply: either (C₁) $\|s\|_{i+1}^* < B(\rho_i)$; or (C₂) s can be split at a *pivot* s_p , i.e. $s = s_1 \cdot s_p \cdot s_2$, such that: (P₁) $\|s_1\|_{i+1}^* < B(\rho_i)$; and (P₂) one *not* ignored place of s_p , the $i+1$'s say, contains more than $B(\rho_i)$ tokens. As a consequence of (P₂) we may ignore the contents of the $i+1$'s place for any covering path for s_{cov} from s_p and thus we can replace s_2 by a path s'_2 with length at most ρ_i . The recurrence relation is then established by noting that a path s to which case (C₁) applies and a path s_1 of case (C₂) (which satisfies property (P₁)) may be replaced by a path of length no more than the covering diameter d_{i+1} of the set $\{(s(1), s(|s|)) : s \text{ path}, \|s\|_{i+1}^* < B(\rho_i)\}$. This yields the *Rackoff recurrence relation* $\rho_{i+1} \leq d_{i+1} + \rho_i$.

It is challenging to apply Rackoff's method to NNCT and we are forced to adjust the above argument in a few technical details. In the setting of NNCT we introduce two families of norms $\|-\|_{\mathcal{N}_i}$ and $\|-\|_{\mathcal{N}_i;C}$ on $Config^\infty$. The norm $\|-\|_{\mathcal{N}_i;C}$ ignores the simple places $p_{(i+1)}, \dots, p_{(n_s)}$ and the norm $\|-\|_{\mathcal{N}_i}$ also ignores coloured tokens and complex places, i.e. for $s \in Config^\infty$ we define $\|s\|_{\mathcal{N}_i} = \max\{|s(p_{(j)})| : j \leq i\}$ and $\|s\|_{\mathcal{N}_i;C} = \max(|s(p'_{(j)})|, \max_{m \in s(p'_{(j)})} \|m\|_{col} : j \in \langle n_c \rangle) \cup \{\|s\|_{\mathcal{N}_i}\}$ where we further define the norm $\|-\|_{col}$ on complex tokens by $\|m\|_{col} = \max\{|m(p'_{(j)})| : j \in \langle n_{col} \rangle\}$.

A. Establishing the Rackoff Recurrence Relation

First, we define bounds that are our key ingredients of $B(\rho_i)$:

$$R := \max(\{|I_r(p)|, |O_r(p)| \mid r \in \mathcal{R}, p \in \mathcal{P}^s \cup \mathcal{P}^c\} \cup \{1\}), \\ R' := R + 1 + \max(\Xi \cup \{\|s_{cov}\|_{\mathcal{N}_s;C}\}) \text{ where} \\ \Xi := \{\|c\|_{col} \mid c \in O_r(p'), p' \in \mathcal{P}^c, r \in SimpleRules\}$$

Lemma 3. Suppose s is a covering path for s_{cov} in \mathcal{N}_{i+1} and $|s(1)(p_{(i+1)})| \geq R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$. Then $\exists s'$ a covering path for s_{cov} in \mathcal{N}_{i+1} such that $s'(1) = s(1)$ and $|s'| \leq \rho_{\mathcal{N}_i}(s_{cov})$.

We then establish that covering paths fall into two categories.

Lemma 4. For all covering paths s for s_{cov} in \mathcal{N}_{i+1} either:

- (C₁) $\|s\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$; or
- (C₂) $s = s_1 \cdot s_p \cdot s_2$ such that $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$ and $\|s_p\|_{\mathcal{N}_{i+1}} \geq R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$.

Unfortunately, we need stronger guarantees on the pivot configuration s_p . We require that there exists a “small” predecessor $s_{p'}$ of s_p , i.e. $\|s_{p'}\|_{\mathcal{N}_{i+1};C} < R' \cdot (\rho_{\mathcal{N}_i}(s_{cov}) + 1)$, that is covered by s_1 . Of course, this does not hold for all pivot configurations, however, we can construct a pivot with this property. We exploit property (P₂) and two observations: let s_0 be a path with $|s| \leq L$ then (O₁) along s_0 only L complex tokens can be moved/removed; and (O₂) for any complex token m and occurring in $s_0(1)$ at most $L \cdot R'$ carried coloured tokens of a given colour can be ejected and removed along s_0 .

Lemma 5. Suppose s is a covering path for s_{cov} in \mathcal{N}_{i+1} . If $L \in \mathbb{N}$ such that $|s| \leq L$ and $\|s(1)\|_{\mathcal{N}_{i+1}} < L \cdot R'$ then there exists a covering path s'' for s_{cov} in \mathcal{N}_{i+1} such that $s''(1) \leq_{Config} s(1)$, $\|s''(1)\|_{\mathcal{N}_{i+1};C} < L \cdot R'$, and $|s''| \leq L$.

Superfluous complex and coloured tokens can thus be removed along $s_1(|s_1|) \cdot s_p \dots$ to strengthen Lemma 4:

Corollary 1. For all covering paths s for s_{cov} in \mathcal{N}_{i+1} either (C'₁) $\|s\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$; or (C'₂) there exist paths s_1 and $s_{p'} \cdot s_2$ such that s_1 is a covering path for $s_{p'}$, $s_1(1) = s(1)$ and $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$; $s_{p'} \cdot s_2$ is a covering path for s_{cov} and $|s_2| \leq \rho_{\mathcal{N}_i}(s_{cov})$; and $\|s_{p'}\|_{\mathcal{N}_{i+1};C} \leq R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$.

Let us write $B_i = R' \cdot \rho_{\mathcal{N}_i}(s_{cov})$, define $P_{(i,B)}$ to be the set of paths of norm bounded by B , i.e. $P_{(i,B)} = \{s : \|s\|_{\mathcal{N}_i}^* < B\}$, and $S_{(i,B)}$ to be the set of pairs (s, s') for which there exists a covering path in $P_{(i,B)}$ from s for s' , i.e. $S_{(i,B)} = \{(s(1), s') : s \in P_{(i,B)}, \|s'\|_{\mathcal{N}_i;C} \leq B, s' \leq_{Config} s(|s|)\}$. Inspecting case (C'₂) we notice that $(s_1(1), s_{p'}) \in S_{(i,B_i)}$. Hence, we can find a path s'_1 of length at most $d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)})$ to replace s_1 . We can thus establish the Rackoff recurrence relation for NNCT:

- Proposition 4.** (i) $\rho_{\mathcal{N}_0}(s_{cov}) \leq d_{\mathcal{N}_0}(S_{(0,R')})$
- (ii) $\rho_{\mathcal{N}_{i+1}}(s_{cov}) \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)}) + \rho_{\mathcal{N}_i}(s_{cov})$.

B. The Covering Diameter of Bounded Paths

The problem is thus reduced to bounding $d_{\mathcal{N}_i}(S_{(i,B)})$. For this purpose we expose a function $\alpha_{i,B}$ and a Petri net $\mathcal{V}_{i,B}$ such that $\alpha_{i,B}$ maps configurations of \mathcal{N}_i to configurations of $\mathcal{V}_{i,B}$.

Definition 7. A *Petri net* is a tuple $\mathcal{V} = (d, F)$ where $d \in \mathbb{N}^+$ and $F \subseteq \mathbb{Z}^d \times \mathbb{N}^d$. For $s, s' \in \mathbb{N}^d$ and $(A, B) \in F$ we have $s \xrightarrow{(A,B)}_{\mathcal{V}} s'$ if $s \geq B$, $s' = s + A$ and $s' \in \mathbb{N}^d$.

We can think of $\mathcal{V}_{i,B}$ as a counter abstraction on \mathcal{N}_i which preserves covering paths (restricted to $P_{(i,B)}$) and their lengths. A complex token m that appears along $s \in P_{(i,B)}$ cannot carry more than B tokens of a particular colour col if m 's col -coloured tokens are ejected along s . This would lead to more than B simple tokens in a configuration along s violating $\|s\|_{\mathcal{N}_i}^* < B$. If m 's col -coloured tokens are *not* ejected along s then $m(col)$ is immaterial and may be abstracted. Hence, for paths in $P_{(i,B)}$, we represent a complex token m as a

map \hat{m} of type $\mathcal{P}^{col} \rightarrow \{0, 1, \dots, B-1, \infty\}$. Note that there are $(B+1)^{n_{col}}$ maps of this type. In $\mathcal{V}_{i,B}$ we keep a counter for each (p', \hat{m}) representing the number of ‘abstract complex tokens’ \hat{m} in complex place p' in addition to i counters representing simple places. Petri net rules can simulate the simple and complex rules of \mathcal{N}_i along paths in $P_{(i,B)}$ and since the number of ‘abstract complex tokens’ is finite we can also simulate transfer transitions as normal Petri net rules. In the interest of readability we relegate the technical definitions of $\mathcal{V}_{i,B}$ and $\alpha_{i,B}$ to the appendix and summarise their properties:

Theorem 5. *For all $i \leq n_s$, $B \in \mathbb{N}$ there exists a Petri net $\mathcal{V}_{i,B} = (d_{i,B}, F_{i,B})$ and a function $\alpha_{i,B}$ such that*

- (A1) $d_{i,B} \leq i + (n_c + 1) \times (B+1)^{n_{col}}$,
- (A2) $R \geq \max\{r(i) \mid r \in F_{i,B}\}$, and
- (A3) for all $s, s' \in S_{i,B}$: $R' \geq \max_{j \in (d_{i,B})} (\alpha_{i,B}(s)(j))$,
- (A4) $\text{dist}_{\mathcal{N}_i}(s, s') \leq \text{dist}_{\mathcal{V}_{i,B}}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$.

Example 2. Suppose we have a NNCT \mathcal{N} with $\mathcal{P}^s = \{p\}$, $\mathcal{P}^c = \{p'\}$, $\mathcal{P}^{col} = \{g, b\}$, $\zeta = [c \mapsto p \mid c \in \mathcal{P}^{col}]$, a complex rule r and a transfer rule r' :

$$r = ((p', \emptyset), (p', [b \mapsto [\bullet]], \emptyset)), \quad r' = ((p', \emptyset), (p', \{b\}, \emptyset)),$$

i.e. both rules r and r' take a complex token from p' to p' ; while doing so r injects a b -token and r' ejects all b -tokens to p . Consider the following configuration s of \mathcal{N} :

$$s = [p \mapsto [\bullet^5], p' \mapsto [m]] \text{ where } m = [g \mapsto [\bullet], b \mapsto [\bullet^{10}]].$$

In $\mathcal{V}_{1,2}$ we have a counter $j_{p',g=1,b \geq 2}$ that represents complex tokens m in p' with $|m(g)| = 1$ and $|m(b)| \geq 2$. We represent s as the configuration $\alpha_{1,2}(s) = [j_p \mapsto 5, j_{p',g=1,b \geq 2} \mapsto 1]$ of $\mathcal{V}_{1,2}$. The rules of $\mathcal{V}_{1,2}$ maintain this representation. For example, for the complex rule r there is a family of rules \hat{r}_{m_0} in $\mathcal{V}_{1,2}$ indexed by ‘abstract complex tokens’. One of them, \hat{r}_{m_0} say, is enabled at $\alpha_{1,2}(s)$ and removes a token from $j_{p',g=1,b \geq 2}$ and adds a token to $j_{p',g=1,b \geq 2}$. Since r injects a b -token and $j_{p',g=1,b \geq 2}$ represents complex token with more than 2 b -tokens, firing \hat{r}_{m_0} does not change the configuration. To illustrate how transfer rules are encoded, consider a different complex token $m' = [g \mapsto [\bullet], b \mapsto [\bullet]]$. In $\mathcal{V}_{1,2}$, r' leads to a family of the rules $\hat{r}'_{m'}$, one of which removes a token from $j_{p',g=1,b=1}$, adds one token to $j_{p',g=1,b=0}$, and adds one token to j_p . Note, the transfer rule r' cannot apply along a path in $P_{(1,2)}$ to a complex token m'_0 with $|m'_0(b)| \geq 2$. Thus, for such ‘too large’ complex tokens no rule $\hat{r}'_{m'_0}$ exists in $\mathcal{V}_{1,2}$. We overcome the mismatch between $\leq_{\mathbb{N}^{d_{i,B}}}$ and \leq_{Config} by adding rules that allow transitions $\alpha_{i,B}([p' \mapsto [m]]) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}([p' \mapsto [m']])$ if $m >_{\mathcal{N}^{col}} m'$.

Since $\mathcal{V}_{i,B}$ is a Petri net, we can appeal to a result on bounds on cover radii, due to Bonnet et al., that shows that (A1)–(A3) are enough to control the covering radius $\rho_{\mathcal{V}_{i,B}}$.

Lemma 6 ([3, Lemma 12 specialised to Petri nets]). Let $\mathcal{V} = (d, F)$ be a Petri net, M_{cov} be the marking to be covered, and $R_{\mathcal{V}} = \max\{|\mathbf{A}(i)|, |\mathbf{B}(i)| : (\mathbf{A}, \mathbf{B}) \in F, i \in \langle d \rangle\}$. If $R'_{\mathcal{V}} = \max_{i \in \langle d \rangle} (M_{\text{cov}}(i), R_{\mathcal{V}})$ then $\rho_{\mathcal{V}}(M_{\text{cov}}) \leq (6R_{\mathcal{V}}R'_{\mathcal{V}})^{(d+1)!}$.

Since $\alpha_{i,B}$ is an expansive map (A4), Lemma 6 immediately gives us a concrete bound for $d_{\mathcal{N}_i}(S_{(i,B)})$:

Corollary 2. Let $i \leq n_s$, $B \in \mathbb{N}$. Then

$$\begin{aligned} d_{\mathcal{N}_i}(S_{(i,B)}) &\leq \max \left\{ \rho_{\mathcal{V}_{i,B}}(\alpha_{i,B}(s')) : \|s'\|_{\mathcal{N}_i;C} \leq B \right\} \\ &\leq (6 \max \{R, B, 1\} \max \{R', B\})^{(\hat{d}_{i,B}+1)!}. \end{aligned}$$

Thus we obtain a concrete bound on the covering radius for s_{cov} by solving the Rackoff recurrence relation.

Theorem 6. *Let us write slog, super-logarithm, for the inverse of $2 \uparrow (-)$, tetration, i.e. $n = 2 \uparrow \text{slog}(n)$. Then for all $i \leq n_s$:*

- (i) $\rho_{\mathcal{N}_0}(s_{\text{cov}}) \leq 2 \uparrow 2 \text{slog}(48n_c n_s n_{\text{col}} R')$
- (ii) $\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) \leq 2^{2^{((\max\{\rho_{\mathcal{N}_i}(s_{\text{cov}}), 2\})^{48n_{\text{col}} n_s n_c R'})}}$ and
- (iii) $\rho_{\mathcal{N}}(s_{\text{cov}}) \leq 2 \uparrow 2n_s + 2 \text{slog}(48(n_s + 1)n_{\text{col}} n_s n_c R')$.

Corollary 3. NNCT Coverability is decidable and in TOWER.

V. A LOWER BOUND

In this section, we show that simple coverability for total-transfer NNCT is TOWER-hard. We encode a bounded counter machine in NNCT which is constructed inductively. Given $n \geq 1$, we construct the *yardstick* counters c_1, \dots, c_n : each counter c_i is bounded by $2 \uparrow i$, and can be incremented, decremented and tested for zero; furthermore operations of the counter c_{i+1} are implemented using operations of the counters c_1, \dots, c_i . Following Lazic et al. [32] we present our proof using pseudo code rather than explicit NNCT rules, which we believe is clearer and more readable. We use a type of Stockmeyer yardstick construction [43] that is reminiscent of Lipton’s EXPSPACE-hardness proof for coverability and reachability for VAS [34]. A $2 \uparrow i$ -bounded counter c is represented by two places: p_c and its *complement place* \bar{p}_c . A valuation $v(c)$ of c is represented by p_c containing $v(c)$ tokens and \bar{p}_c containing $2 \uparrow i - v(c)$ tokens. Places p_c and \bar{p}_c maintain the invariant that the number of tokens they carry sum up to $2 \uparrow i$ at all times. An increment (decrement) of c is then implemented by adding a token to p_c (\bar{p}_c) and removing one from \bar{p}_c (p_c). For a zero test $\text{iszero}(c)$ an additional $2 \uparrow i$ -bounded counter s_i is maintained. On the invocation of $\text{iszero}(c)$ a non-deterministic number k of tokens are removed from \bar{p}_c and k is added to s_i , which is assumed to be 0 at the invocation of $\text{iszero}(c)$. The operation $\text{ismax\&reset}(-)$ is then applied to s_i which performs a decrement of precisely $2 \uparrow i$ on s_i and blocks if $s_i < 2 \uparrow i$. This of course means that $\text{ismax\&reset}(-)$ can only succeed if $c = 0$ to begin with and $k = 2 \uparrow i$. Our construction is similar to Lazic’s proof of TOWER-hardness for VAS with one stack [30]. Lazic uses the $2 \uparrow i$ -bounded counters to enumerate all possible stacks over a binary alphabet of height $2 \uparrow i$ while decrementing the given counter one-by-one. In the case of NNCT stacks are not available, however, we can encode arrays into complex tokens in a yardstick fashion; and, instead of enumerating stacks, we enumerate all binary arrays of length $2 \uparrow i$.

Theorem 7. *Simple coverability, boundedness and termination for total-transfer NNCT is TOWER-hard.*

Proof. We deduce TOWER-hardness by showing that a polynomial-time computable NNCT \mathcal{N}_M can weakly bisimulate a deterministic bounded two-counter machine \mathcal{M} , of size n and $2 \uparrow n$ -bounded counters. The machine \mathcal{M} supports the operations: $x++$, $x--$, $reset(x)$, $iszero(x)$, and $ismax(x)$.

Each simulation state of \mathcal{N}_M represents a valuation v of $6n + 2$ active and inactive counters, and n arrays. In addition to the counters x , y of \mathcal{M} , the NNCT \mathcal{N}_M simulates the auxiliary counters s_i , p_i , p'_i , c_i , c'_i , and an auxiliary array a_i for each $i \leq n$. To simplify notation, we write C_i for the set of counters $\{s_i, p_i, p'_i, c_i, c'_i\}$ when $i < n$, and C_n for the set of counters $\{s_n, p_n, p'_n, c_n, c'_n, x, y\}$. Each active counter $d \in C_i$ is $2 \uparrow i$ -bounded, each inactive counter has an undefined value. Each array a_i has length $(2 \uparrow i) + 1$ and carries values $a_i(j) \in \{0, 1, 2\}$ for $j \leq 2 \uparrow i$. The NNCT \mathcal{N}_M has two simple places $p[d]$ and $\bar{p}[d]$ for each counter $d \in C_i$, three complex places $p[a_i]$, $\bar{p}[a_i]$, and $p[aux_i]$, two colours $p[j_i]$ and $\bar{p}[j_i]$ for each array a_i , a complex “sink” place $p[disc]$, and ζ maps $p[j_i]$ to $p[p'_i]$ and $\bar{p}[j_i]$ to $\bar{p}[p'_i]$. Lastly, \mathcal{N}_M has a polynomial number of simple places encoding the control of \mathcal{M} and the internal control of \mathcal{N}_M . Further, \mathcal{N}_M 's transfer rules are all total, and hence \mathcal{N}_M is a total-transfer NNCT.

A valuation v is represented by a configuration s as follows:

- For each i and $d \in C_i$, if d is active then there are exactly $v(d)$ \bullet -tokens in $p[d]$ and $2 \uparrow i - v(d)$ \bullet -tokens in $\bar{p}[d]$.
- For each inactive counter d , $p[d]$ and $\bar{p}[d]$ are empty.
- Each array a_i is represented as follows. For $k \leq 2 \uparrow i$, let $m_{(i,k)}$ be a complex token such that $m_{(i,k)}$ contains exclusively tokens of colours $p[j_i]$ and $\bar{p}[j_i]$: k tokens of colour $p[j_i]$ and $2 \uparrow i - k$ tokens of colour $\bar{p}[j_i]$. Then, to represent a_i , there are exactly $v(a_i(k))$ tokens $m_{(i,k)}$ in $p[a_i]$ and $2 - v(a_i(k))$ tokens $m_{(i,k)}$ in $\bar{p}[a_i]$.
- For each i the place $p[aux_i]$ is empty.

The question whether \mathcal{M} reaches a halting control state from its initial state can then be answered by performing a simple coverability query on \mathcal{N}_M 's simple places encoding \mathcal{M} 's finite control. Assuming that only \mathcal{M} 's halting control states have no successors, \mathcal{M} 's halting problem also reduces to the termination problem for \mathcal{N}_M . Augmenting \mathcal{N}_M with an additional simple place that is incremented with every transition shows that \mathcal{M} 's halting problem reduces to boundedness of \mathcal{N}_M .

We implement further instructions to improve readability. The NNCT \mathcal{N}_M simulates $activate(d)$, $deactivate(d)$ for $d \in \bigcup_{i \in \langle n \rangle} C_i$, and operations $reset(d)$, $iszero(d)$, and $ismax(d)$ for $d \in \bigcup_{i \in \langle n \rangle} C_i \setminus \{s_i\}$. For each $i \in \langle n \rangle$, we implement $ismax \& reset(s_i)$ and the specialised operations:

$isequal(p_i, p'_i)$, $a_i(p_i)++$, $a_i(p_i)--$, $reset(a_i(p_i))$,
 $iszero(a_i(p_i))$, $ismax(a_i(p_i))$, $activate(a_i)$.

These operations only succeed if the counters in question are active. The counters in C_1 are 2-bounded so implementing operations on them is trivial. For $i < n$, operations on a_i are simulated using counters in C_i and operations on counters in C_{i+1} are simulated using operations on counters in C_i and a_i .

(i) *The implementation of $ismax \& reset(s_{i+1})$:*

```
for  $p_i := 0$  to  $2 \uparrow i$  do ( $reset(a_i(p_i))$ );
while ( $iszero(a_i(2 \uparrow i))$ ) do
   $s_{i+1}--$ ;  $reset(p_i)$ ;  $a_i(p_i)++$ ;
  while  $ismax(a_i(p_i))$  do ( $a_i(p_i)--$ ;  $p_i++$ ;  $a_i(p_i)++$ );
```

After the for-loop, we know that $a_i(x) = 0$ for all $x \leq 2 \uparrow i$. The array a_i is the binary representation of a number between 0 and $2 \uparrow (i + 1)$. This number is initially 0 and the outer while loop performs a long addition of 1 for each iteration. If $v(a_i(v(p_i))) = 2$ then $v(p_i)$ is an index representing a carry bit. For each number represented by a_i we perform $s_{i+1}--$. Hence, if initially $v(s_{i+1}) = 2 \uparrow (i + 1)$, then after performing $ismax \& reset(s_{i+1})$ the resulting v' sets $v'(s_{i+1}) = 0$. If $v(s_{i+1}) < 2 \uparrow (i + 1)$, then after $v(s_{i+1})$ iterations the resulting valuation v' sets $v'(s_{i+1}) = 0$ and a_i represents the number $v(s_{i+1})$. Since $v(s_{i+1}) < 2 \uparrow (i + 1)$ this implies that $a_i(2 \uparrow i) = 0$ and hence the body of the outer while loop is executed again leading to an invocation of $s_{i+1}--$ which blocks. Hence $ismax \& reset(s_{i+1})$ blocks when executed in a configuration representing v such that $v(s_{i+1}) < 2 \uparrow (i + 1)$.

The implementation of $ismax \& reset(s_{i+1})$ assumes that operations on a_i are correctly implemented. In the following we show how $a_i(p_i)++$ is simulated which is representative.

(ii) *The implementation of $a_i(p_i)++$:*

```
Move a complex token from  $\bar{p}[a_i]$  to  $p[aux_i]$ ;
deactivate( $p'_i$ );
Eject the contents of a complex token in  $p[aux_i]$  and place
its remains into  $p[disc]$ ;
isequal( $p_i, p'_i$ );  $reset(p'_i)$ ;
Create an empty complex token in  $p[aux_i]$ ;
while ( $p_i \neq p'_i$ ) do ( $p'_i++$ );
  Inject a  $p[j_i]$ -coloured token into a complex token in  $p[aux_i]$ ;
while ( $\neg(ismax(p'_i))$ ) do ( $p'_i++$ );
  Inject a  $\bar{p}[j_i]$ -coloured token into a complex token in  $p[aux_i]$ ;
Move a complex token from  $p[aux_i]$  to  $p[a_i]$ ;  $reset(p'_i)$ ;
```

Suppose $a_i(p_i)++$ is executed in a configuration s that represents valuation v and p_i, p'_i are active. If $v(a_i(v(p_i))) < 2$ then there exists a complex token $m_{(i,v(p_i))}$ in $\bar{p}[a_i]$ and all complex tokens in $\bar{p}[a_i]$ are of the form $m_{(i,k)}$ for some $k \leq 2 \uparrow i$. The move of one $m_{(i,k)}$ complex token from $\bar{p}[a_i]$ to $p[aux_i]$ results in $p[aux_i]$ containing just $m_{(i,k)}$, since by assumption $p[aux_i]$ is empty before. After deactivating p'_i , both $p[p'_i]$ and $\bar{p}[p'_i]$ are empty. Ejecting the contents of $m_{(i,k)}$ removes $m_{(i,k)}$ from $p[aux_i]$, inserts k \bullet -tokens into $p[p'_i]$ and $2 \uparrow i - k$ \bullet -tokens into $\bar{p}[p'_i]$, and places the remaining empty complex token into $p[disc]$. Disregarding a_i , the configuration we have reached represents a partial valuation v' that sets $v'(p'_i) = k$ and $v'(p_i) = v(p_i)$. After executing $isequal(p_i, p'_i)$, the simulation only succeeds if $k = v(p_i)$. Hence $\bar{p}[a_i]$ now contains $2 - (v(a_i(v(p_i))) + 1)$ complex tokens $m_{(i,v(p_i))}$ and the same number of other complex tokens $m_{(i,k)}$. The two while loops carefully inject $p[j_i]$ and $\bar{p}[j_i]$ -coloured tokens into the newly created token at $p[aux_i]$ to yield a new $m_{(i,v(p_i))}$ located in $p[aux_i]$ which we move to $p[a_i]$. Thus, $p[a_i]$ now contains $v(a_i(v(p_i))) + 1$ complex tokens $m_{(i,v(p_i))}$ and the

same number of other complex tokens $m_{(i,k)}$ as before. The configuration s' we have reached thus represents a valuation v'' such that $v''(a_i(j)) = v(a_i(j))$ for all $j \leq 2 \uparrow i$ and $j \neq v(p_i)$ and $v''(a_i(v(p_i))) = v(a_i(v(p_i))) + 1$. Otherwise, if $v(a_i(v(p_i))) = 2$, the simulation either blocks on the attempt to move some $m_{(i,k)}$ from $\bar{p}[a_i]$ to $p[aux_i]$ or on the execution of $isequal(p_i, p'_i)$ since it is impossible to obtain $k = v(p_i)$.

(iii) *The implementation of $iszero(d)$:*

```
while (*) do ( $d++$ ;  $s_{i+1}++$ ;);  $ismax\&reset(s_{i+1})$ ;  
while (*) do ( $d--$ ;  $s_{i+1}++$ ;);  $ismax\&reset(s_{i+1})$ 
```

Note that $s_{i+1} = 0$ is a precondition for $iszero(d)$ and we only modify s_{i+1} in $iszero(d)$ and $ismax\&reset(s_{i+1})$. Thus, except when executing $iszero(d)$, we maintain $s_{i+1} = 0$.

If $ismax\&reset(s_{i+1})$ completes after the first while loop, then the loop must have incremented s_{i+1} and d to $2 \uparrow (i + 1)$. Since d is $2 \uparrow (i + 1)$ -bounded, d had to be 0 to begin with and is valued $2 \uparrow (i + 1)$ after $ismax\&reset(s_{i+1})$'s first invocation. The rest of the implementation then guarantees that d is decremented to 0 again. Hence $iszero(d)$ succeeds only if started in a configuration that values d as 0.

(iv) We omit the analogous implementations of the other operations here. We refer the reader to Appendix D.

The initial operation of $\mathcal{N}_{\mathcal{M}}$ executes $activate(-)$ for all counters and arrays in turn, in order of the bound size and length. From this point on, $\mathcal{N}_{\mathcal{M}}$ weakly bisimulates \mathcal{M} . Thus, we can reduce the halting problem for \mathcal{M} to simple coverability, termination, or boundedness of $\mathcal{N}_{\mathcal{M}}$. \square

VI. RELATED WORK

Asynchronously Communicating Pushdown Systems: In 2006, Sen and Viswanathan showed that safety verification is decidable for recursive asynchronous programs [42], which give rise to ACPS that satisfy the empty-stack constraint. Liveness properties [23] and practical analyses [27] for asynchronous programs have since been studied. Recently, Ganty and Majumdar showed that a variety of verification problems for asynchronous programs are polynomial-time inter-reducible to decision problems on Petri nets, thus e.g. safety verification is EXPSpace-complete. Extensions of Sen and Viswanathan's model [8, 7, 15] have been proposed that allow e.g. the modelling of higher-order stack automata, the dynamic creation of task buffers, or WQO stack alphabets; however, the empty-stack restriction remains a key restriction. The empty-stack constraint fits nicely with the atomicity requirement on asynchronous procedures. An atomic procedure only needs to make synchronisations before and after its execution which may thus be performed with an empty call stack. The shaped-constraint enables a relaxation of the atomicity requirement: it allows non-trivial synchronisations inside the execution of procedure call. This increase in expressive power, together with the ramping up of the computational complexity, confirms our intuition that shaped-stack ACPS are a much more general model than ACPS satisfying the empty-stack constraint.

Concurrent Pushdown Systems (CPS): Numerous classes with decidable verification problems have been discovered: parallel flow graph systems [18], visibly pushdown

automata with FIFO-channels [1], CPS communicating over locks [28], recursive programs with hierarchical communication [6, 4], and CPS with FIFO-channels for which the empty-stack restriction applies to sends [26]. Further, over-approximation [21, 25] and under-approximation techniques [17, 36, 5, 44] have been studied. Czerwinski et al. introduced PCCFG to study bisimulation for BPC [9], a process algebra extending BPA and BPP [16]. However, synchronisation between processes (which transforms PCCFG to APCPS) is not a feature considered by Czerwinski et al.

Extensions of Petri nets: Coverability is a central decision problem in the vast literature of Petri net extensions. However, any non-trivial extension, such as *reset arcs* or *transfer arcs* [14, 41], typically renders coverability non-primitive recursive. *Nested Petri nets* may appear closely related to NNCT, they are however much more expressive and coverability is Ackermann-hard [35]. Nested Petri nets allow arbitrary nesting of tokens, and nested layers of a token can synchronise. By contrast, internal synchronisation is not possible in NNCT: coloured tokens can only be ejected to simple places and then inspected. Our proof exploits this fact and it seems to explain the TOWER-membership of NNCT coverability.

Data nets [32] allow tokens to be drawn from an arbitrary linearly ordered set. Recently, coverability and termination for data nets and a subclass, Petri data nets (PDN), were shown to be $\mathbf{F}_{\omega^{\omega^{\omega}}}$ -complete [24, 40]. A more restricted subclass of PDN studied by Lazic et al. gives rise to a TOWER-hard coverability problem, namely, *unordered* PDN (UPDN) which features an equality check on tokens. Lazic et al. show that coverability, termination and boundedness are all TOWER-hard, but no upper-bound is available. Unfortunately, the equality check on tokens makes it unclear whether coverability of UPDN reduces to NNCT coverability which is why we opted for a TOWER-hardness proof from first principles. Adding the ability to create fresh tokens to UPDNs yields ν -Petri nets (ν -PN) for which coverability is ACK-hard [39].

Rackoff technique. Originally introduced to show the EXPSpace-membership of coverability and boundedness for VAS [37], the *Rackoff technique* has recently become popular. It has been used to establish EXPSpace upper bounds for coverability and boundedness of *strongly increasing affine nets* (SIAN) [3], selective unboundedness of VAS with states (VASS) [10], model-checking Petri nets [2], an ALTSPACE upper bound for coverability and boundedness for branching VAS [11], and a TOWER-upper bound for a coverability problem for alternating BVAS with states (ABVASS) [31]. Even though NNCT coverability and ABVASS coverability are TOWER-complete, it is not obvious how to inter-reduce the coverability problems between NNCT and ABVASS. It is hard to see how one can simulate in an ABVASS the ability of NNCT to model complex tokens, carrying an unbounded number of coloured tokens, that can interact via ejection with other complex tokens. In the other direction there is no clear counterpart to the tree-like runs of ABVASS in NNCT.

Vector addition systems with one stack (SVAS). Recent work has shown that boundedness and termination are decidable

for SVAS and that the problem lies in HACK [33]. The decidability of coverability and reachability is still an open question but are known to be TOWER-hard [30].

Future Directions: We intend to identify a practical coverability algorithm for shaped ACPS, as they arise from the abstract interpretation of realistic Erlang programs [12, 13].

Acknowledgments: Financial support by EPSRC (grant EP/F036361/1 and DTG doctoral studentship for the first author) is gratefully acknowledged. We thank Matthew Hague, Stefan Kiefer, James Worrell, Javier Esparza, Sylvain Schmitz, Ranko Lazic, and Alain Finkel for insightful comments.

REFERENCES

- [1] D. Babic and Z. Rakamaric. Asynchronously communicating visibly pushdown systems. In *FMOODS/FORTE*, pages 225–241, 2013.
- [2] M. Blockelet and S. Schmitz. Model Checking Coverability Graphs of Vector Addition Systems. In *MFCS*, pages 108–119, 2011.
- [3] R. Bonnet, A. Finkel, and M. Praveen. Extending the Rackoff technique to Affine nets. In *FSTTCS*, pages 301–312, 2012.
- [4] A. Bouajjani and M. Emmi. Analysis of Recursively Parallel Programs. In *POPL*, pages 203–214, 2012.
- [5] A. Bouajjani and M. Emmi. Bounded Phase Analysis of Message-Passing Programs. In *TACAS*, pages 451–465, 2012.
- [6] A. Bouajjani, M. Müller-Olm, and T. Touili. Regular Symbolic Analysis of Dynamic Networks of Pushdown Systems. In *CONCUR*, pages 473–487, 2005.
- [7] X. Cai and M. Ogawa. Well-Structured Pushdown Systems. In *CONCUR*, pages 121–136, 2013.
- [8] R. Chadha and M. Viswanathan. Decidability Results for Well-Structured Transition Systems with Auxiliary Storage. In *CONCUR*, pages 136–150, 2007.
- [9] W. Czerwinski, S. B. Fröschle, and S. Lasota. Partially-Commutative Context-Free Processes. In *CONCUR*, pages 259–273, 2009.
- [10] S. Demri. On Selective Unboundedness of VASS. *J. Comput. Syst. Sci.*, 79(5):689–713, 2013.
- [11] S. Demri, M. Jurdzinski, O. Lachish, and R. Lazic. The Covering and Boundedness Problems for Branching Vector Addition Systems. In *FSTTCS*, pages 181–192, 2009.
- [12] E. D’Osualdo, J. Kochems, and C.-H. L. Ong. Soter: An Automatic Safety Verifier for Erlang. In *AGERE! ’12*, pages 137–140, 2012.
- [13] E. D’Osualdo, J. Kochems, and C.-H. L. Ong. Automatic Verification of Erlang-Style Concurrency. In *SAS*, pages 454–476, 2013.
- [14] C. Dufourd, A. Finkel, and P. Schnoebelen. Reset Nets Between Decidability and Undecidability. In *ICALP*, pages 103–115, 1998.
- [15] M. Emmi, P. Ganty, and R. Majumdar. An Expressive yet Decidable Model of Asynchronous Event-Driven Programs. Preprint, 2014.
- [16] J. Esparza. Petri Nets, Commutative Context-Free Grammars, and Basic Parallel Processes. *Fundam. Inform.*, 31(1):13–25, 1997.
- [17] J. Esparza and P. Ganty. Complexity of Pattern-Based Verification for Multithreaded Programs. In *POPL*, pages 499–510, 2011.
- [18] J. Esparza and A. Podolski. Efficient Algorithms for pre* and post* on Interprocedural Parallel Flow Graphs. In *POPL*, pages 1–11, 2000.
- [19] J. Esparza, P. Ganty, S. Kiefer, and M. Luttenberger. Parikh’s Theorem: A Simple and Direct Automaton Construction. *Inf. Process. Lett.*, 111(12):614–619, 2011.
- [20] A. Finkel and P. Schnoebelen. Well-Structured Transition Systems Everywhere! *TCS*, 256(1-2):63–92, 2001.
- [21] C. Flanagan and S. Qadeer. Thread-Modular Model Checking. In *SPIN*, pages 213–224, 2003.
- [22] P. Ganty and R. Majumdar. Algorithmic Verification of Asynchronous Programs. *ACM TOPLAS*, 34(1):6, 2012.
- [23] P. Ganty, R. Majumdar, and A. Rybalchenko. Verifying Liveness for Asynchronous Programs. In *POPL*, pages 102–113, 2009.
- [24] S. Haddad, S. Schmitz, and P. Schnoebelen. The Ordinal-Recursive Complexity of Timed-arc Petri Nets, Data Nets, and Other Enriched Nets. In *LICS*, pages 355–364, 2012.
- [25] T. A. Henzinger, R. Jhala, R. Majumdar, and S. Qadeer. Thread-Modular Abstraction Refinement. In *CAV*, pages 262–274, 2003.
- [26] A. Heußner, J. Leroux, A. Muscholl, and G. Sutre. Reachability Analysis of Communicating Pushdown Systems. In *FOSSACS*, pages 267–281, 2010.
- [27] R. Jhala and R. Majumdar. Interprocedural Analysis of Asynchronous Programs. In *POPL*, pages 339–350, 2007.
- [28] V. Kahlon. Boundedness vs. Unboundedness of Lock Chains: Characterizing Decidability of Pairwise CFL-Reachability for Threads Communicating via Locks. In *LICS*, pages 27–36, 2009.
- [29] J. Kochems and C.-H. L. Ong. Safety Verification of Asynchronous Pushdown Systems with Shaped Stacks. In *CONCUR*, pages 288–302, 2013.
- [30] R. Lazic. The Reachability Problem for Vector Addition Systems with a Stack Is Not Elementary. *CoRR*, abs/1310.1767, 2013.
- [31] R. Lazic and S. Schmitz. Non-Elementary Complexities for Branching VASS, MELL, and Extensions. In *CSL/LICS*, pages 61:1–61:10, 2014.
- [32] R. Lazic, T. C. Newcomb, J. Ouaknine, A. W. Roscoe, and J. Worrell. Nets with Tokens Which Carry Data. In *ICATPN*, pages 301–320, 2007.
- [33] J. Leroux, M. Praveen, and G. Sutre. Hyper-Ackermannian Bounds for Pushdown Vector Addition Systems. In *CSL/LICS*, pages 63:1–63:10, 2014.
- [34] R. J. Lipton. The Reachability Problem Requires Exponential Space. Technical report, Dept. of Computer Science, Yale, 1976.
- [35] I. A. Lomazova and P. Schnoebelen. Some Decidability Results for Nested Petri Nets. In *Ershov Memorial Conf.*, pages 208–220, 1999.
- [36] S. Qadeer and J. Rehof. Context-Bounded Model Checking of Concurrent Software. In *TACAS*, pages 93–107, 2005.
- [37] C. Rackoff. The Covering and Boundedness Problems for Vector Addition Systems. *TCS*, 6:223–231, 1978.
- [38] G. Ramalingam. Context-Sensitive Synchronization-Sensitive Analysis Is Undecidable. *ACM TOPLAS*, 22(2):416–430, 2000.
- [39] F. Rosa-Velardo and D. de Frutos-Escrig. Decidability and Complexity of Petri Nets with Unordered Data. *TCS*, 412(34):4439–4451, 2011.
- [40] S. Schmitz. Complexity Hierarchies Beyond Elementary. *CoRR*, abs/1312.5686, 2013.
- [41] P. Schnoebelen. Revisiting Ackermann-Hardness for Lossy Counter Machines and Reset Petri Nets. In *MFCS*, pages 616–628, 2010.
- [42] K. Sen and M. Viswanathan. Model Checking Multithreaded Programs with Asynchronous Atomic Methods. In *CAV*, pages 300–314, 2006.
- [43] L. J. Stockmeyer. *The Complexity of Decision Problems in Automata Theory and Logic*. PhD thesis, MIT, 1974.
- [44] S. L. Torre and M. Napoli. Reachability of Multistack Pushdown Systems with Scope-Bounded Matching Relations. In *CONCUR*, pages 203–218, 2011.

A. Proofs for Section II

Lemma 1. *Coverability and simple coverability for ACPS polynomial-time inter-reduce.*

Proof. The reduction from simple coverability to coverability of ACPS is trivial since the former is a subproblem of the latter. For the other direction suppose we have a coverability query $Q = (\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}})$. Let us assume that $\mathcal{P} = (\mathcal{Q}, \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R})$, $\Pi_{\text{cov}} = (q_1, \beta_1) \parallel \dots \parallel (q_n, \beta_n)$ and $\Pi_0 = (q_1^0, \beta_1^0) \parallel \dots \parallel (q_m^0, \beta_m^0)$ where $q_i, q_j^0 \in \mathcal{Q}$ and $\beta_i, \beta_j^0 \in \mathcal{A}^*$ for $i \in \langle n \rangle, j \in \langle m \rangle$.

Let us define the following ACPS $\mathcal{P}' = (\mathcal{Q}', \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R}')$ where $\mathcal{Q}' = \mathcal{Q} \cup \{q'_{(i,A,\beta')}, q'_{(i,\epsilon)}, q'_{(j,\beta'')}, \beta_i = A \cdot \beta \cdot \beta', \beta_j = \beta'' \cdot \beta''', \beta_k = \epsilon, i, k \in \langle n \rangle, j \in \langle m \rangle\}$ and all control states $q'_{(i,A,\beta)}, q'_{(k,\epsilon)}, q'_{(j,\beta)}$ are fresh and

$\mathcal{R}' = \mathcal{R}$

$$\cup \left\{ \begin{array}{l} (q'_{(i,A,\beta')}, B) \xrightarrow{\epsilon} (q'_{(i,A,\beta')}, \epsilon), \quad \left| \begin{array}{l} \beta_i = A \beta'_i, \\ \beta'_i = \beta_0 \cdot B \cdot \beta', \\ \beta'_i = \beta_1 \cdot \beta'', \\ i \in \langle n \rangle \end{array} \right. \\ (q'_{(i,A,\beta'')}, B') \xrightarrow{\epsilon} (q'_{(i,\beta'')}, \epsilon), \\ (q_i, A) \xrightarrow{\epsilon} (q'_{(i,A,\beta'_i)}, \epsilon) \end{array} \right\}$$

$$\cup \left\{ \begin{array}{l} (q_k, \epsilon) \xrightarrow{\epsilon} (q'_{(k,\epsilon)}, \epsilon), \quad \left| \begin{array}{l} \beta_k = \epsilon, \\ \beta_j^0 = \beta''' \cdot C \cdot \beta'''' \\ k \in \langle n \rangle, j \in \langle m \rangle \end{array} \right. \\ (q_{(j,\beta''')}, D_0) \xrightarrow{\epsilon} (q'_{(j,\beta''')}, D_0 C), \\ (q_{(j,\epsilon)}, D_0) \xrightarrow{\epsilon} (q'_{(j,\epsilon)}, \epsilon) \end{array} \right\}$$

The ACPS \mathcal{P}' essentially implements the query Q' . The rules involving $q'_{(j,\beta)}$ set up the start configuration with arbitrary stacks from a length one stack. Rules involving $q'_{(k,\epsilon)}$ and $q'_{(i,A,\epsilon)}$ essentially check that the coverability query is satisfied. In order to account for this we change the coverability query to $Q' = (\mathcal{P}, \Pi'_0 \triangleleft \Gamma_0, \Pi'_{\text{cov}} \triangleleft \Gamma_{\text{cov}})$ where $\Pi'_0 = (q'_{(1,\beta_1^0)}, D_0) \parallel \dots \parallel (q'_{(n,\beta_n^0)}, D_0)$, $\Pi'_{\text{cov}} = (q'_{(1,\tilde{\beta}_1)}, \epsilon) \parallel \dots \parallel (q'_{(n,\tilde{\beta}_n)}, \epsilon)$ and if $\beta_i = \epsilon$ then $\tilde{\beta}_1 = \epsilon$; otherwise if $\beta_i = A \cdot \beta'$ then $\tilde{\beta}_i = A, \epsilon$. By construction Q' is a simple query.

Since any set of suffixes/prefixes of a sequence β satisfies

$$|\{\beta_0 : \beta_0 \cdot \beta_1 = \beta\}| = |\{\beta_1 : \beta_0 \cdot \beta_1 = \beta\}| \leq |\beta|$$

we can clearly set that $|\mathcal{Q}'| \leq |\mathcal{Q}| + (n+m) \times \max\{|\beta_i|, |\beta_j^0| : i \in \langle n \rangle, j \in \langle m \rangle\}$ and $|\mathcal{R}'| \leq |\mathcal{R}| + 3(n+m) \times \max\{|\beta_i|, |\beta_j^0| : i \in \langle n \rangle, j \in \langle m \rangle\}$ and hence \mathcal{P}' and Q' are clearly polynomial-time computable from \mathcal{P} and Q .

By construction \mathcal{P}' has the following property: $(q'_{(i,A,\beta_1)}, A\beta) \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{*} (q'_{(i,A,\epsilon)}, \epsilon) \parallel \Pi_0 \triangleleft \Gamma$ if and only if $\beta_i = A\beta_0\beta_1, \beta_1 \leq_{\mathcal{A}^*} \beta$, i.e. checking coverability at a process level is correctly implemented by each process. Further $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*} \Pi \triangleleft \Gamma$ if, and only if, $\Pi'_0 \triangleleft \Gamma_0 \xrightarrow{*} \Pi \triangleleft \Gamma$, i.e. Π'_0 correctly sets up Π_0 .

Suppose now that Q is a yes-instance. This means that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*} \Pi \triangleleft \Gamma$ such that $\Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}} \leq_{\text{ACPS}} \Pi \triangleleft \Gamma$. Clearly it is then the case that $\Pi'_0 \triangleleft \Gamma_0 \xrightarrow{*} \Pi \triangleleft \Gamma$. Since $\Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}} \leq_{\text{ACPS}} \Pi \triangleleft \Gamma$ we know that $\Pi = (q_1, \beta'_1) \parallel \dots \parallel$

$(q_n, \beta'_n) \parallel \Pi'$ and for all $i \in \langle n \rangle$ either $\beta_i = \epsilon$ or $\beta_i = A \cdot \beta''_i$, $\beta'_i = A \cdot \beta'''_i$ and $\beta'_i \leq_{\mathcal{A}^*} \beta'''_i$.

And hence $\Pi'_0 \triangleleft \Gamma_0 \xrightarrow{*} \Pi \triangleleft \Gamma \xrightarrow{*} (q'_{(1,\tilde{\beta}_1)}, \epsilon) \parallel \dots \parallel (q'_{(n,\tilde{\beta}_n)}, \epsilon) \parallel \Pi' \triangleleft \Gamma$ and hence Q' is a yes-instance for coverability.

For the other direction, suppose Q' is a yes-instance for coverability. We then know that $\Pi'_0 \triangleleft \Gamma_0 \xrightarrow{*} (q'_{(1,\tilde{\beta}_1)}, \beta'_1) \parallel \dots \parallel (q'_{(n,\tilde{\beta}_n)}, \beta'_n) \parallel \Pi' \triangleleft \Gamma' =: s'$ such that

$$(q'_{(1,\tilde{\beta}_1)}, \epsilon) \parallel \dots \parallel (q'_{(n,\tilde{\beta}_1)}, \epsilon) \triangleleft \Gamma \leq_{\text{ACPS}} s'.$$

Then our observation above tells us that it must be the case that (possibly reordering locally-independent transitions) $\Pi'_0 \triangleleft \Gamma_0 \xrightarrow{*} (q_1, \beta''_1 \beta'_1) \parallel \dots \parallel (q_n, \beta''_n \beta'_n) \parallel \Pi' \triangleleft \Gamma' \xrightarrow{*} (q'_{(1,\beta_1)}, \beta''_1 \beta'_1) \parallel \dots \parallel (q'_{(n,\beta_n)}, \beta''_n \beta'_n) \parallel \Pi' \triangleleft \Gamma'$ with either $\beta_i = \epsilon$ or $\beta_i = A\beta_i^\dagger, \beta''_i = A\beta'''_i$ and $\beta_i^\dagger \leq_{\mathcal{A}^*} \beta'''_i$ and thus $\beta_i^\dagger \leq_{\mathcal{A}^*} \beta'''_i \beta'_i$ for all $i \in \langle n \rangle$. Further $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*} (q_1, \beta''_1 \beta'_1) \parallel \dots \parallel (q_n, \beta''_n \beta'_n) \parallel \Pi' \triangleleft \Gamma'$ Hence Q is a yes-instance for coverability.

We can thus conclude that simple coverability and coverability are polynomial-time inter-reducible. \square

1) *Proof of Proposition 1:* In this section we will give a proof of the following Proposition:

Proposition 1. *Given an ACPS \mathcal{P} , a simple coverability query Q and a $\Pi^0 \triangleleft \Gamma^0$ there exists ACPS $\mathcal{F}(\mathcal{P})$ in normal form, a simple coverability query $\mathcal{F}(Q)$, and $\mathcal{F}(\Pi^0 \triangleleft \Gamma^0)$ — all polynomial-time computable — such that: Q is a yes-instance if, and only if, $\mathcal{F}(Q)$ is a yes-instance; and \mathcal{P} is bounded (terminating) from $\Pi^0 \triangleleft \Gamma^0$ if, and only if, $\mathcal{F}(\mathcal{P})$ is bounded (terminating respectively) from $\mathcal{F}(\Pi^0 \triangleleft \Gamma^0)$.*

We will give a proof in two steps: (i) we first transform a general ACPS \mathcal{P} into an ACPS that satisfies a *pre-normal form* as defined below; (ii) secondly, we show how to transform an ACPS in pre-normal form with the desired property. We lay out our argument in the two Lemmas below, but first we define *pre-normal form*: We say an ACPS $\mathcal{P} = (\mathcal{Q}, \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R})$ is in pre-normal if for all $(q, \beta) \xrightarrow{\lambda} (q', \beta')$ (i') if $\lambda \neq \epsilon$ then $\beta = \beta' = \epsilon$, otherwise either (ii') $\beta = A \in \mathcal{A}$ and $\beta' = \epsilon$ or (iii') $\beta = \epsilon$ and $\beta' = A' \in \mathcal{A}$; or (iv') $\beta = \beta' = \epsilon$; and (v') if $\lambda = \nu(q'', \beta'')$ then $\beta'' = \epsilon$.

Lemma 7. *Given an ACPS \mathcal{P} , a simple coverability query Q and a start configuration $\Pi^0 \triangleleft \Gamma^0$ there exists ACPS $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ in pre-normal form, simple coverability query $\mathcal{F}^{\text{pnf}}(Q)$, and start configuration $\mathcal{F}^{\text{pnf}}(\Pi^0 \triangleleft \Gamma^0)$ — all polynomial-time computable — such that: (A) Q is a yes-instance if, and only if, $\mathcal{F}^{\text{pnf}}(Q)$ is a yes-instance; (B) \mathcal{P} is bounded from $\Pi^0 \triangleleft \Gamma^0$ if, and only if, $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ is bounded from $\mathcal{F}^{\text{pnf}}(\Pi^0 \triangleleft \Gamma^0)$; and (C) \mathcal{P} is terminating from $\Pi^0 \triangleleft \Gamma^0$ if, and only if, $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ is terminating from $\mathcal{F}^{\text{pnf}}(\Pi^0 \triangleleft \Gamma^0)$.*

Proof. Let us fix an ACPS $\mathcal{P} = (\mathcal{Q}, \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R})$ and

let us define the ACPS $\mathcal{P}_0 = (\mathcal{Q}', \mathcal{A}, Chan, Msg, \mathcal{R}')$ and

$$\begin{aligned} \mathcal{Q}' &= \mathcal{Q} \cup \left\{ q_{\beta_0}^{\text{pop}}, q_{\beta_1}^{\text{push}} \mid (q, \beta) \xrightarrow{\lambda} (q', \beta') \in \mathcal{R}', \beta = \beta_0' \cdot \beta_0, \right. \\ &\quad \left. \beta' = \beta_1 \cdot \beta_1' \right\} \\ &\cup \left\{ q_{\beta_0''}^{\text{push}} \mid (q, \beta) \xrightarrow{\nu(q'', \beta'')} (q', \beta') \in \mathcal{R}' \right. \\ &\quad \left. \beta'' = \beta_0'' \cdot \beta_1'' \right\} \\ &\cup \{q^{\text{cov}} \mid q \in \mathcal{Q}\} \\ \mathcal{R}' &= \left\{ (q_{\beta}^{\text{pop}}, \epsilon) \xrightarrow{\lambda'} (q_{\beta'}^{\text{push}}, \epsilon) \mid \begin{array}{l} (q, \beta) \xrightarrow{\lambda} (q', \beta') \in \mathcal{R} \\ \text{if } \lambda = \nu(q'', \beta'') \\ \text{then } \lambda' = \nu(q_{\beta''}^{\text{push}}, \epsilon) \\ \text{otherwise } \lambda' = \lambda \end{array} \right\} \\ &\cup \left\{ (q, \epsilon) \xrightarrow{\epsilon} (q_{\epsilon}^{\text{pop}}, \epsilon), \right. \\ &\quad \left. (q_{\epsilon}^{\text{push}}, \epsilon) \xrightarrow{\epsilon} (q', \epsilon) \mid (q, \beta) \xrightarrow{\lambda} (q', \beta') \in \mathcal{R}' \right\} \\ &\cup \left\{ (q_{\beta}^{\text{pop}}, A) \xrightarrow{\epsilon} (q_{\beta \cdot A}^{\text{pop}}, \epsilon), \right. \\ &\quad \left. (q_{\beta' \cdot A'}^{\text{push}}, \epsilon) \xrightarrow{\epsilon} (q_{\beta'}^{\text{push}}, A'), \mid q_{\beta}^{\text{pop}}, q_{\beta \cdot A}^{\text{pop}} \in \mathcal{Q}' \right. \\ &\quad \left. q_{\beta' \cdot A'}^{\text{push}}, q_{\beta'}^{\text{push}} \in \mathcal{Q}' \right\} \end{aligned}$$

The rules of \mathcal{P}_0 simply implement a $(q, \beta) \xrightarrow{\lambda} (q', \beta')$ by popping β one symbol at the time and then pushing β' one symbol at the time. It is easy to see that \mathcal{P}_0 is in pre-normal form. Further let $\Xi = \{|\beta|, |\beta'| : (q, \beta) \xrightarrow{\lambda} (q', \beta') \in \mathcal{R}'\} \cup \{|\beta''| : (q, \beta) \xrightarrow{\nu(q'', \beta'')} (q', \beta') \in \mathcal{R}'\}$ then $|\mathcal{Q}'| \leq 2|\mathcal{Q}| + 3 \times |\mathcal{R}| \times \max(\Xi)$ and $|\mathcal{R}'| \leq 2 \times |\mathcal{Q}'| + 3 \times |\mathcal{R}|$ and so \mathcal{P}_0 is clearly polynomial-time computable from \mathcal{P} .

We will show that there is a *weak reflexive bisimulation* between \mathcal{P} and \mathcal{P}_0 .

Definition (Weak reflexive bisimulation). Suppose (S, \xrightarrow{u}_S) and $(S', \xrightarrow{u}_{S'})$ are labelled transition systems we say a relation $\mathcal{B} \subseteq S \times S'$ is a *weak reflexive simulation* if for all $(s, s') \in \mathcal{B}$, if for some $t \in S$ we have $s \xrightarrow{u}_S t$ then either $(t, s') \in \mathcal{B}$ and $u = \epsilon$ or there exists $t' \in S'$ such that $s' \xrightarrow{u}_{S'}^* t'$ and $(t, t') \in \mathcal{B}$. We say \mathcal{B} is a *weak reflexive bisimulation* relation just if both \mathcal{B} and \mathcal{B}^{-1} are weak reflexive simulation relations.

We temporarily label the transition systems $(\mathcal{P}, \rightarrow_{\mathcal{P}})$ and $(\mathcal{P}_0, \rightarrow_{\mathcal{P}_0})$ with rules of \mathcal{R} . Let us label the transition $\Pi \triangleleft \Gamma \xrightarrow{r}_{\mathcal{P}} \Pi' \triangleleft \Gamma'$ if the rule $r \in \mathcal{R}$ is used to justify the transition. We label \mathcal{P}_0 's transition as follows: If $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}_0} \Pi' \triangleleft \Gamma'$ using a rule $(q_{\beta}^{\text{pop}}, \epsilon) \xrightarrow{\lambda'} (q_{\beta'}^{\text{push}}, \epsilon) \in \mathcal{R}'$ introduced because of a rule $r = (q, \beta) \xrightarrow{\lambda} (q', \beta') \in \mathcal{R}$ we label the transition $\Pi \triangleleft \Gamma \xrightarrow{r}_{\mathcal{P}_0} \Pi' \triangleleft \Gamma'$; otherwise we label the transition with ϵ , i.e. $\Pi \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}_0} \Pi' \triangleleft \Gamma'$.

Let first define a representation function for the state of a pushdown process:

$$F(q, \beta) = \{(q, \beta)\} \cup \left\{ (q_{\beta_1}^{\text{push}}, \beta_2 \cdot \beta''') \mid \begin{array}{l} \exists (q', \beta') \xrightarrow{\lambda} (q, \beta'') \in \mathcal{R} \\ \beta = \beta'' \cdot \beta''' \\ \beta'' = \beta_1 \cdot \beta_2 \end{array} \right\}$$

$$\cup \left\{ (q_{\beta_1}^{\text{push}}, \beta_2 \cdot \beta''') \mid \begin{array}{l} \exists (q_0, \beta_0) \xrightarrow{\lambda} (q_1, \beta_1) \in \mathcal{R} \\ \lambda = \nu(q, \beta) \\ \beta = \beta_1 \cdot \beta_2 \end{array} \right\} \cup \left\{ (q_{\beta_1}^{\text{pop}}, \beta_2 \cdot \beta''') \mid \begin{array}{l} \exists (q, \beta') \xrightarrow{\lambda} (q', \beta'') \in \mathcal{R} \\ \beta = \beta' \cdot \beta''' \\ \beta' = \beta_1 \cdot \beta_2 \end{array} \right\}$$

with which we can now relate configurations of \mathcal{P} and \mathcal{P}_0 using the relation: $\mathcal{B} = \{(\Pi \triangleleft \Gamma, \Pi' \triangleleft \Gamma') : \Pi = \pi_1 \parallel \dots \parallel \pi_n, \Pi' = \pi'_1 \parallel \dots \parallel \pi'_n, \forall i \in \langle n \rangle. \pi'_i \in F(\pi_i)\}$.

Let us now prove that \mathcal{B} is a weak reflexive simulation. Suppose $(\pi \parallel \Pi \triangleleft \Gamma, \pi_0 \parallel \Pi_0 \triangleleft \Gamma) \in \mathcal{B}$, and clearly $\pi = (q, \beta)$, and $(q, \beta) \parallel \Pi \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}} (q', \beta') \parallel \Pi' \triangleleft \Gamma'$. Clearly this must happen using rule $l = (q, \beta_0) \xrightarrow{\lambda} (q', \beta'_0)$, and, $\beta = \beta_0 \cdot \beta_1$ and $\beta' = \beta'_0 \cdot \beta_1$. We will briefly show that we can assume $\pi_0 = (q_{\beta_0}^{\text{pop}}, \beta_1)$. We observe that we can then perform the following ϵ -labelled transitions:

$$\begin{aligned} (q_{\beta''}^{\text{push}}, \beta''') \parallel \Pi_0 \triangleleft \Gamma &\xrightarrow{\epsilon}_{\mathcal{P}_0} (q_{\epsilon}^{\text{push}}, \beta'' \cdot \beta''') \parallel \Pi_0 \triangleleft \Gamma \\ &\xrightarrow{\epsilon}_{\mathcal{P}_0} (q_{\epsilon}, \beta'' \cdot \beta''') \parallel \Pi_0 \triangleleft \Gamma \\ &= (q_{\epsilon}, \beta) \parallel \Pi_0 \triangleleft \Gamma \\ &\xrightarrow{\epsilon}_{\mathcal{P}_0} (q_{\epsilon}^{\text{pop}}, \beta) \parallel \Pi_0 \triangleleft \Gamma \\ &= (q_{\epsilon}^{\text{pop}}, \beta_0 \cdot \beta_1) \parallel \Pi_0 \triangleleft \Gamma \\ &\xrightarrow{\epsilon}_{\mathcal{P}_0} (q_{\beta_0}^{\text{pop}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma \end{aligned}$$

it should be clear that for all $\pi_0 \in F(q, \beta)$ we can ϵ -transition $\pi_0 \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}_0} (q_{\beta_0}^{\text{pop}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma$. Hence we will assume in the following that $\pi_0 = (q_{\beta_0}^{\text{pop}}, \beta_1)$.

First we note that there is a rule $(q_{\beta_0}^{\text{pop}}, \epsilon) \xrightarrow{\lambda'} (q_{\beta_0'}^{\text{push}}, \epsilon) \in \mathcal{R}'$. We can thus make a case analysis on λ .

- *Case:* $\lambda = \epsilon$.

Then clearly $\Pi = \Pi'$ and $\Gamma' = \Gamma$ and $\lambda' = \epsilon$ and:

$$(q_{\beta_0}^{\text{pop}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}_0} (q_{\beta_0'}^{\text{push}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma.$$

Thus we can see that $(q_{\beta_0'}^{\text{push}}, \beta_1) \in F(q', \beta')$ and hence $((q', \beta') \parallel \Pi' \triangleleft \Gamma, (q_{\beta_0'}^{\text{push}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $\lambda = c ! m$.

Then clearly $\Pi = \Pi'$ and $\Gamma' = \Gamma \oplus [c \mapsto [m]]$ and $\lambda' = c ! m$ and:

$$(q_{\beta_0}^{\text{pop}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}_0} (q_{\beta_0'}^{\text{push}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma'.$$

Thus we can see that $(q_{\beta_0'}^{\text{push}}, \beta_1) \in F(q', \beta')$ and hence $((q', \beta') \parallel \Pi' \triangleleft \Gamma', (q_{\beta_0'}^{\text{push}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma') \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $\lambda = c ? m$.

Then clearly $\Pi = \Pi'$ and $\Gamma = \Gamma' \oplus [c \mapsto [m]]$ and $\lambda' = c ? m$ and:

$$(q_{\beta_0}^{\text{pop}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}_0} (q_{\beta_0'}^{\text{push}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma'.$$

Thus we can see that $(q_{\beta_0'}^{\text{push}}, \beta_1) \in F(q', \beta')$ and hence

$((q', \beta') \parallel \Pi' \triangleleft \Gamma', (q_{\beta'_0}^{\text{push}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma') \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $\lambda = \nu(q'', \beta'')$.

Then clearly $\Pi' = (q'', \beta'') \parallel \Pi$ and $\Gamma' = \Gamma$ and $\lambda' = \nu(q_{\beta''}^{\text{push}}, \epsilon)$ and:

$$(q_{\beta_0}^{\text{pop}}, \beta_1) \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}_0} (q_{\beta'_0}^{\text{push}}, \beta_1) \parallel (q_{\beta''}^{\text{push}}, \epsilon) \parallel \Pi_0 \triangleleft \Gamma'.$$

Thus we can see that $(q_{\beta'_0}^{\text{push}}, \beta_1) \in F(q', \beta')$, $(q_{\beta''}^{\text{push}}, \epsilon) \in F(q'', \beta'')$ and hence $((q', \beta') \parallel \Pi' \triangleleft \Gamma', (q_{\beta'_0}^{\text{push}}, \beta_1) \parallel (q_{\beta''}^{\text{push}}, \epsilon) \parallel \Pi_0 \triangleleft \Gamma') \in \mathcal{B}$ which is what we wanted to prove.

Hence we can conclude that \mathcal{B} is a weak reflexive simulation.

Let us turn now to \mathcal{B}^{-1} . Suppose $(\pi \parallel \Pi \triangleleft \Gamma, \pi_0 \parallel \Pi_0 \triangleleft \Gamma) \in \mathcal{B}$ and $\pi_0 \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}} \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0$ using rule $r \in \mathcal{R}'$.

Let us perform a case analysis on r

- *Case:* $r = (q, \epsilon) \xrightarrow{\epsilon} (q_{\epsilon}^{\text{pop}}, \epsilon)$.

Then clearly $\pi_0 = (q, \beta)$, $\pi = (q, \beta)$ and $\pi'_0 = (q_{\epsilon}^{\text{pop}}, \beta)$, $\Pi'_0 = \Pi_0$, and $\Gamma'_0 = \Gamma$. Further $(q_{\epsilon}^{\text{pop}}, \beta) \in F(q, \beta)$ and thus $(\pi \parallel \Pi \triangleleft \Gamma, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $r = (q_{\epsilon}^{\text{push}}, \epsilon) \xrightarrow{\epsilon} (q, \epsilon)$.

Clearly $\pi_0 = (q_{\epsilon}^{\text{push}}, \beta)$, $\pi = (q, \beta)$ and $\pi'_0 = (q, \beta)$, $\Pi'_0 = \Pi_0$, and $\Gamma'_0 = \Gamma$. Further $(q, \beta) \in F(q, \beta)$ and thus $(\pi \parallel \Pi \triangleleft \Gamma, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $r = (q_{\beta}^{\text{pop}}, A) \xrightarrow{\epsilon} (q_{\beta \cdot A}^{\text{pop}}, \epsilon)$.

Clearly $\pi_0 = (q_{\beta}^{\text{pop}}, A \cdot \beta')$, $\pi = (q, \beta \cdot A \cdot \beta')$ and $\pi'_0 = (q_{\beta \cdot A}^{\text{pop}}, \beta')$, $\Pi'_0 = \Pi_0$, and $\Gamma'_0 = \Gamma$. Further $(q_{\beta \cdot A}^{\text{pop}}, \beta') \in F(q, \beta \cdot A \cdot \beta')$ and thus $(\pi \parallel \Pi \triangleleft \Gamma, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $r = (q_{\beta \cdot A}^{\text{push}}, \epsilon) \xrightarrow{\epsilon} (q_{\beta}^{\text{push}}, A)$.

Clearly $\pi_0 = (q_{\beta \cdot A}^{\text{push}}, \beta')$, $\pi = (q, \beta \cdot A \cdot \beta')$ and $\pi'_0 = (q_{\beta}^{\text{push}}, A \cdot \beta')$, $\Pi'_0 = \Pi_0$, and $\Gamma'_0 = \Gamma$. Further $(q_{\beta}^{\text{push}}, \beta') \in F(q, \beta \cdot A \cdot \beta')$ and thus $(\pi \parallel \Pi \triangleleft \Gamma, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $r = (q_{\beta}^{\text{pop}}, \epsilon) \xrightarrow{\lambda} (q_{\beta'}^{\text{push}}, \epsilon)$.

Clearly $\pi_0 = (q_{\beta}^{\text{pop}}, \beta'')$, $\pi = (q, \beta \cdot \beta'')$ and $\pi'_0 = (q_{\beta'}^{\text{push}}, \beta'')$. Further there is a rule $(q, \beta) \xrightarrow{\lambda'} (q', \beta') \in \mathcal{R}$.

Let us do case analysis on λ

- *Case:* $\lambda = \epsilon$.

Then $\Pi'_0 = \Pi_0$, $\Gamma'_0 = \Gamma$ and $\lambda' = \epsilon$. And $(q, \beta \beta'') \parallel \Pi \triangleleft \Gamma \xrightarrow{r}_{\mathcal{P}} (q', \beta' \beta'') \parallel \Pi \triangleleft \Gamma$. Clearly $(q_{\beta'}^{\text{push}}, \beta'') \in F(q', \beta' \beta'')$ and thus $((q', \beta' \beta'') \parallel \Pi \triangleleft \Gamma, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $\lambda = c!m$

Then $\Pi'_0 = \Pi_0$, $\Gamma'_0 = \Gamma \oplus [c \mapsto [m]]$ and $\lambda' = c!m$. And $(q, \beta \beta'') \parallel \Pi \triangleleft \Gamma \xrightarrow{r}_{\mathcal{P}} (q', \beta' \beta'') \parallel \Pi \triangleleft \Gamma'_0$. Clearly $(q_{\beta'}^{\text{push}}, \beta'') \in F(q', \beta' \beta'')$ and thus $((q', \beta' \beta'') \parallel \Pi \triangleleft \Gamma'_0, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $\lambda = c?m$

Then $\Pi'_0 = \Pi_0$, $\Gamma = \Gamma'_0 \oplus [c \mapsto [m]]$ and $\lambda' = c?m$. And $(q, \beta \beta'') \parallel \Pi \triangleleft \Gamma \xrightarrow{r}_{\mathcal{P}} (q', \beta' \beta'') \parallel \Pi \triangleleft \Gamma'_0$. Clearly $(q_{\beta'}^{\text{push}}, \beta'') \in F(q', \beta' \beta'')$ and thus $((q', \beta' \beta'') \parallel \Pi \triangleleft \Gamma'_0, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

- *Case:* $\lambda = \nu(q_{\beta''}^{\text{push}}, \epsilon)$

Then $\Pi'_0 = (q_{\beta''}^{\text{push}}, \epsilon) \parallel \Pi_0$, $\Gamma = \Gamma'_0$ and $\lambda' = \nu(q'', \beta''')$. And $(q, \beta \beta'') \parallel \Pi \triangleleft \Gamma \xrightarrow{r}_{\mathcal{P}} (q', \beta' \beta'') \parallel (q'', \beta''') \parallel \Pi \triangleleft \Gamma$. Firstly $(q_{\beta'}^{\text{push}}, \beta'') \in F(q', \beta' \beta'')$ and $(q_{\beta''}^{\text{push}}, \epsilon) \in F(q'', \beta''')$. Thus $((q', \beta' \beta'') \parallel (q'', \beta''') \parallel \Pi \triangleleft \Gamma'_0, \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'_0) \in \mathcal{B}$ which is what we wanted to prove.

Hence \mathcal{B}^{-1} is a weak reflexive simulation and hence \mathcal{B} is a weak reflexive bisimulation.

Let us now define $\mathcal{F}^{\text{pnf}}(\mathcal{P}) = (\mathcal{Q}', \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R}' \cup \mathcal{R}_{\text{cov}})$ where

$$\mathcal{R}_{\text{cov}} = \left\{ \begin{array}{l} (q_{\epsilon}^{\text{pop}}, \epsilon) \xrightarrow{\epsilon} (q^{\text{cov}}, \epsilon), \\ (q_{\epsilon}^{\text{push}}, \epsilon) \xrightarrow{\epsilon} (q^{\text{cov}}, \epsilon), \\ (q_{A \cdot \beta}^{\text{pop}}, \epsilon) \xrightarrow{\epsilon} (q^{\text{cov}}, A), \\ (q_{A' \cdot \beta'}^{\text{push}}, \epsilon) \xrightarrow{\epsilon} (q^{\text{cov}}, A'), \\ (q, \epsilon) \xrightarrow{\epsilon} (q^{\text{cov}}, \epsilon), \end{array} \left| \begin{array}{l} q_{\epsilon}^{\text{pop}}, q_{\epsilon'}^{\text{push}} \in \mathcal{Q}', \\ q_{A \cdot \beta}^{\text{pop}}, q_{A' \cdot \beta'}^{\text{push}} \in \mathcal{Q}', \\ q \in \mathcal{Q} \end{array} \right. \right\}.$$

Adding the rules \mathcal{R}_{cov} to \mathcal{P}_0 (which are non-reversible) only changes which configurations are reachable/coverable by a one step transition. Obviously $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ remains polynomial time computable from \mathcal{P}

Suppose that $Q = (\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ is a simple coverability query where $\Pi = (q_1, \beta_1) \parallel \dots \parallel (q_k, \beta_k)$ and $\beta_i \in \mathcal{A} \cup \{\epsilon\}$ then let $\mathcal{F}^{\text{pnf}}(Q) = (\mathcal{F}^{\text{pnf}}(\mathcal{P}), \Pi_0 \triangleleft \Gamma_0, \Pi' \triangleleft \Gamma')$ be a simple coverability query such that $\Pi' = (q_1^{\text{cov}}, \beta_1) \parallel \dots \parallel (q_k^{\text{cov}}, \beta_k)$.

Suppose Q is a yes-instance then $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}} \Pi_1 \triangleleft \Gamma_1$ such that $\Pi \triangleleft \Gamma \leq_{\text{ACPS}} \Pi_1 \triangleleft \Gamma_1$. Since \mathcal{B} is a reflexive weak bisimulation we know $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}_0} \Pi'_1 \triangleleft \Gamma_1$ such that $(\Pi_1 \triangleleft \Gamma_1, \Pi'_1 \triangleleft \Gamma_1) \in \mathcal{B}$. Since $\Pi \triangleleft \Gamma \leq_{\text{ACPS}} \Pi_1 \triangleleft \Gamma_1$ we know that $\Pi_1 = (q_1, \beta'_1) \parallel \dots \parallel (q_k, \beta'_k) \parallel \Pi_2$ such that for all $i \in \langle k \rangle$ either $\beta_i = \epsilon$ or $\beta_i = A_i \in \mathcal{A}$ and $\beta'_1 = A_i \beta'_1$. Hence we can deduce that $\Pi'_1 = \pi'_1 \parallel \dots \parallel \pi'_k \parallel \Pi'_2$ such that $\pi'_i \in F(q_i, \beta'_i)$ for all $i \in \langle k \rangle$. Thus it is easy to see that $\Pi'_1 \triangleleft \Gamma_1 \xrightarrow{*}_{\mathcal{F}^{\text{pnf}}(\mathcal{P})} (q_1^{\text{cov}}, \beta'_1) \parallel \dots \parallel (q_k^{\text{cov}}, \beta'_k) \parallel \Pi'_2 \triangleleft \Gamma_1$ such that for all $i \in \langle k \rangle$ it is the case that $\beta_i'' = A \beta_i'''$ and $\beta'_i = A \beta_i''$ and hence $\mathcal{F}^{\text{pnf}}(Q)$ is a yes-instance.

Conversely, suppose $\mathcal{F}^{\text{pnf}}(Q)$ is a yes instance then $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{F}^{\text{pnf}}(\mathcal{P})} (q_1^{\text{cov}}, \beta'_1) \parallel \dots \parallel (q_k^{\text{cov}}, \beta'_k) \parallel \Pi'_1 \triangleleft \Gamma_1$ such that for all $i \in \langle k \rangle$ either $\beta_i = \epsilon$ or $\beta_i = A_i$ and $\beta'_i = A_i \beta'_i$. Hence clearly (by reversing transitions from \mathcal{R}_{cov}) $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}_0} \pi_1 \parallel \dots \parallel \pi_k \parallel \Pi'_1 \triangleleft \Gamma_1$ where $\pi'_i \in F(q_i, \beta'_i)$ for some $\beta'_1, \dots, \beta'_k$ such that either $\beta_i = \epsilon$ or $\beta'_i = A_i \beta_i''$ for all $i \in \langle k \rangle$. Since \mathcal{B} is a reflexive weak bisimulation we know that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}} (q_1, \beta'_1) \parallel \dots \parallel (q_k, \beta'_k) \parallel \Pi_1 \triangleleft \Gamma_1$. Hence Q is a yes-instance. We can thus conclude that Q is a yes-instance iff $\mathcal{F}^{\text{pnf}}(Q)$ is a yes-instance.

For boundedness, let $\Pi^0 \triangleleft \Gamma^0$ be a start configuration and let $\mathcal{F}^{\text{pnf}}(\Pi^0 \triangleleft \Gamma^0) = \Pi^0 \triangleleft \Gamma^0$. Suppose that $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}} \Pi \triangleleft \Gamma\}$ is a finite set. We notice that for all $\Pi \triangleleft \Gamma$ the set $\{\Pi' \triangleleft \Gamma' : (\Pi \triangleleft \Gamma, \Pi' \triangleleft \Gamma') \in \mathcal{B}\}$ is finite. Thus using that \mathcal{B} is a reflexive weak bisimulation we can infer that $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}_0} \Pi \triangleleft \Gamma\}$ is a finite set. Since the rules in \mathcal{R}_{cov} adds only a finite number of reachable configurations we can conclude $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{F}^{\text{pnf}}(\mathcal{P})} \Pi \triangleleft \Gamma\}$ is a finite set and thus $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ is bounded from $\mathcal{F}^{\text{pnf}}(\Pi^0 \triangleleft \Gamma^0)$.

Conversely, suppose $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{F}^{\text{pnf}}(\mathcal{P})} \Pi \triangleleft \Gamma\}$ is a finite set. Then since the rules in \mathcal{R}_{cov} adds only a finite number of reachable configurations we can infer $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}_0} \Pi \triangleleft \Gamma\}$ is a finite set. We further note: $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}} \Pi \triangleleft \Gamma\} \subseteq \{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \xrightarrow{*}_{\mathcal{P}_0} \Pi \triangleleft \Gamma\}$ and thus \mathcal{P} is bounded from $\Pi_0 \triangleleft \Gamma_0$. Thus \mathcal{P} is bounded from $\Pi_0 \triangleleft \Gamma_0$ if and only if $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ is bounded from $\mathcal{F}^{\text{pnf}}(\Pi_0 \triangleleft \Gamma_0)$.

For termination, let $\Pi_0 \triangleleft \Gamma_0$ be a start configuration and define again $\mathcal{F}^{\text{pnf}}(\Pi^0 \triangleleft \Gamma^0) = \Pi^0 \triangleleft \Gamma^0$. Suppose there exists an infinite path s in \mathcal{P} starting from $\Pi_0 \triangleleft \Gamma_0$. Since \mathcal{B} is a weak reflexive bisimulation and s uses an infinite sequence of labels it is clear that there is a path s' in \mathcal{P}_0 and s' is also an infinite path. The path s' is clearly also a path of $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ hence $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ is non-terminating from $\mathcal{F}^{\text{pnf}}(\Pi^0 \triangleleft \Gamma^0)$. Conversely, suppose that s' is an infinite path in $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ starting from $\Pi_0 \triangleleft \Gamma_0$. Since a path can only be finitely extended by rules in \mathcal{R}_{cov} we can deduce that there is also an infinite path s'' from $\Pi_0 \triangleleft \Gamma_0$ in \mathcal{P}_0 . If s'' gives rise to an infinite sequence of labels then, since \mathcal{B} is a weak reflexive bisimulation, we clearly obtain a path s in \mathcal{P} that is also infinite (since all transitions in \mathcal{P} carry a label). Suppose for a contradiction that s'' gives rise only for a finite sequence of labels. This implies there exists an infinite path s_0 in \mathcal{P}_0 such that for all i the transition $s(i) \xrightarrow{\epsilon}_{\mathcal{P}} s(i+1)$ is an ϵ -transition. Inspecting the definition \mathcal{P}_0 we see this implies all rules used must of the form $(q, \beta) \xrightarrow{\epsilon} (q', \beta')$. We can conclude that this is impossible since there are no cycles in \mathcal{P}_0 's rules with no side effects. Hence we can conclude that \mathcal{P} is non-terminating from $\Pi^0 \triangleleft \Gamma^0$ if, and only if, $\mathcal{F}^{\text{pnf}}(\mathcal{P})$ is non-terminating from $\mathcal{F}^{\text{pnf}}(\Pi_0 \triangleleft \Gamma_0)$. \square

We can now use a summarisation-inspired idea to encode control-states into the stack alphabet:

Lemma 8. Given an ACPS \mathcal{P} in pre-normal form, a simple coverability query Q and a start configuration $\Pi^0 \triangleleft \Gamma^0$ there exists ACPS $\mathcal{F}^{\text{nf}}(\mathcal{P})$ satisfying: for all rules $(q, \beta) \xrightarrow{\lambda} (q', \beta')$ of $\mathcal{F}^{\text{nf}}(\mathcal{P})$ (i) $q = q'$, (ii) $\beta = A \in \mathcal{A}$, (iii) if $\lambda \neq \epsilon$ then $\beta' = A' \in \mathcal{A}$, otherwise (iv) $\beta' \in \{\epsilon, A', BC : A', B, C \in \mathcal{A}\}$, and (v) $\lambda = \nu(q'', \beta)$ then $q'' = q$, and $\beta \in \mathcal{A}$; simple coverability query $\mathcal{F}^{\text{nf}}(Q)$; and start configuration $\mathcal{F}^{\text{nf}}(\Pi^0 \triangleleft \Gamma^0)$ — all polynomial-time computable — such that: (A) Q is a yes-instance if, and only if, $\mathcal{F}^{\text{nf}}(Q)$ is a yes-instance; and (B) \mathcal{P} is bounded (terminating) from $\Pi^0 \triangleleft \Gamma^0$ if, and only if, $\mathcal{F}^{\text{nf}}(\mathcal{P})$

is bounded (terminating respectively) from $\mathcal{F}^{\text{nf}}(\Pi^0 \triangleleft \Gamma^0)$.

Proof. Suppose $\mathcal{P} = (\mathcal{Q}, \mathcal{A}, \text{Chan}, \text{Msg}, \mathcal{R})$ is an ACPS in pre-normal form. We then define $\mathcal{P}' = (\mathcal{Q}', \mathcal{A}', \text{Chan}, \text{Msg}, \mathcal{R}')$ where $\mathcal{Q}' = \{q_0\}$ and q_0 is a fresh control state, $\mathcal{A}' = \{A^{(q, q')} \mid q, q' \in \mathcal{Q}, A \in \mathcal{Q} \cup \{\Theta\}\}$, where Θ is a fresh symbol, and \mathcal{R}' is obtained from \mathcal{R} as follows

$$\mathcal{R}' = \left\{ \begin{array}{l} A^{(q, q'')} \xrightarrow{\lambda'} A^{(q', q'')} \mid \begin{array}{l} (q, \epsilon) \xrightarrow{\lambda} (q', \epsilon) \in \mathcal{R}, \\ A \in \mathcal{A} \cup \{\Theta\}, q'', q'_0 \in \mathcal{Q}, \\ \text{if } \lambda = \nu(q_0, \epsilon) \text{ then } \lambda' = \nu\Theta^{(q_0, q'_0)} \\ \text{otherwise } \lambda = \lambda' \end{array} \end{array} \right\} \\ \cup \left\{ \begin{array}{l} A^{(q, q'')} \xrightarrow{\epsilon} B^{(q', q''')} A^{(q'', q''')} \mid \begin{array}{l} (q, \epsilon) \xrightarrow{\epsilon} (q', B) \in \mathcal{R}, \\ A \in \mathcal{A} \cup \{\Theta\}, \\ q'', q''' \in \mathcal{Q} \end{array} \end{array} \right\} \\ \cup \left\{ A^{(q, q')} \xrightarrow{\epsilon} \epsilon \mid (q, A) \xrightarrow{\epsilon} (q', \epsilon) \in \mathcal{R} \right\}$$

where rather than writing (q_0, A) we just write A . It is easy to see that \mathcal{P}' is polynomial time computable from \mathcal{P} .

We represent the state of a pushdown process by the following function:

$$F(q, A_1 A_2 A_3 \dots A_n) = \left\{ A_1^{(q, q_1)} A_2^{(q_1, q_2)} A_3^{(q_2, q_3)} \dots A_n^{(q_{n-1}, q_n)} \Theta^{q_n, q_{n+1}} \mid \Xi \right\}$$

where $\Xi = q_1, \dots, q_{n+1} \in \mathcal{Q}$. We use the former to represent a \mathcal{P} -configuration as a set:

$$G(\Pi \triangleleft \Gamma) = \left\{ \begin{array}{l} \Pi \triangleleft \Gamma \mid \begin{array}{l} \Pi = \pi_1 \parallel \dots \parallel \pi_n, \\ \Pi' = \pi'_1 \parallel \dots \parallel \pi'_n, \\ \forall i \in \langle n \rangle. \pi'_i \in F(\pi_i) \end{array} \end{array} \right\}.$$

Further we define a relation of configurations and sets of configurations:

$$\mathcal{B} = \{(\Pi \triangleleft \Gamma, G(\Pi \triangleleft \Gamma))\}.$$

Let us define a *co-universal powerset lifting* of the transition system induced by \mathcal{P}' and $\rightarrow_{\mathcal{P}_0}$ as follows $\mathcal{P}[\mathcal{P}'] := (\mathcal{P}[\mathcal{M}[\mathcal{Q}' \times \mathcal{A}']], \rightarrow_{\mathcal{P}[\mathcal{P}]})$ where $S \rightarrow_{\mathcal{P}[\mathcal{P}']} S'$ just if for all $s' \in S'$ there exists $s \in S$ such that $s \rightarrow_{\mathcal{P}_0} s'$. Further we temporarily label \mathcal{P} and $\mathcal{P}[\mathcal{P}']$ by \mathcal{P} -configurations as follows: if $s \rightarrow_{\mathcal{P}} s'$ we label by s' the transition $s \xrightarrow{s'}_{\mathcal{P}} s'$. If $S \rightarrow_{\mathcal{P}[\mathcal{P}']} S'$ such that $S' = G(s')$ for some \mathcal{P} -configuration s' we label by s' the transition $S \xrightarrow{s'}_{\mathcal{P}[\mathcal{P}']} S'$.

We will show that \mathcal{B} is a bisimulation relation for \mathcal{P} and $\mathcal{P}[\mathcal{P}']$. As a first step let us give a proof that \mathcal{B} is a simulation relation: Suppose $(\pi \parallel \Pi \triangleleft \Gamma, G(\pi \parallel \Pi \triangleleft \Gamma)) \in \mathcal{B}$ and $\pi \parallel \Pi \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}} \pi' \parallel \Pi' \triangleleft \Gamma'$ using rule $r \in \mathcal{R}$. We know that $l = \pi' \parallel \Pi' \triangleleft \Gamma'$. So let $\pi'_1 \parallel \Pi'_1 \triangleleft \Gamma' \in G(\pi' \parallel \Pi' \triangleleft \Gamma')$ such that $\Pi'_1 \triangleleft \Gamma' \in G(\Pi' \triangleleft \Gamma')$ and $\pi'_1 \in F(\pi')$. Let us perform a case analysis on r :

- *Case:* $r = (q, \epsilon) \xrightarrow{\epsilon} (q', B)$.

In this case $\pi = (q, A_1 \dots A_n)$, $\pi' = (q', B A_1 \dots A_n)$, $\Gamma = \Gamma'$ and $\Pi = \Pi'$. From this we can deduce:

$\pi'_1 = B^{(q', q_1)} A_1^{(q_1, q_2)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_{n+1}, q_{n+2})}$ for some q_1, \dots, q_{n+2} . Let $\pi_1 = A_1^{(q, q_2)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_{n+1}, q_{n+2})}$ then clearly $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \rightarrow_{\mathcal{P}_0} \pi'_1 \parallel \Pi'_1 \triangleleft \Gamma$ using rule $A_1^{(q, q_2)} \xrightarrow{\epsilon} B^{(q', q_1)} A_1^{(q_1, q_2)}$. Further $\pi_1 \in F(\pi)$ and hence $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \in G(\pi \parallel \Pi' \triangleleft \Gamma) = G(\pi \parallel \Pi \triangleleft \Gamma)$. And so since $\pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ is an arbitrary element of $G(\pi' \parallel \Pi' \triangleleft \Gamma')$ we can deduce $G(\pi \parallel \Pi \triangleleft \Gamma) \xrightarrow{l_{\mathcal{P}[\mathcal{P}']}} G(\pi' \parallel \Pi' \triangleleft \Gamma')$. and $(\pi' \parallel \Pi' \triangleleft \Gamma', G(\pi' \parallel \Pi' \triangleleft \Gamma')) \in \mathcal{B}$.

- *Case:* $r = (q, A) \xrightarrow{\epsilon} (q', \epsilon)$.

In this case $\pi = (q, A A_1 \dots A_n)$, $\pi' = (q', A_1 \dots A_n)$, $\Gamma = \Gamma'$ and $\Pi = \Pi'$. From this we can deduce: $\pi'_1 = A_1^{(q', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ for some q_1, \dots, q_{n+1} . Let $\pi_1 = A_1^{(q, q')} A_1^{(q', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ then clearly $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \rightarrow_{\mathcal{P}_0} \pi'_1 \parallel \Pi'_1 \triangleleft \Gamma$ using rule $A_1^{(q, q')} \xrightarrow{\epsilon} \epsilon$. Further $\pi_1 \in F(\pi)$ and hence $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \in G(\pi \parallel \Pi' \triangleleft \Gamma) = G(\pi \parallel \Pi \triangleleft \Gamma)$. And so since $\pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ is an arbitrary element of $G(\pi' \parallel \Pi' \triangleleft \Gamma')$ we can deduce $G(\pi \parallel \Pi \triangleleft \Gamma) \xrightarrow{l_{\mathcal{P}[\mathcal{P}']}} G(\pi' \parallel \Pi' \triangleleft \Gamma')$. and $(\pi' \parallel \Pi' \triangleleft \Gamma', G(\pi' \parallel \Pi' \triangleleft \Gamma')) \in \mathcal{B}$.

- *Case:* $r = (q, \epsilon) \xrightarrow{\lambda} (q', \epsilon)$.

In this case $\pi = (q, A_1 \dots A_n)$, $\pi' = (q', A_1 \dots A_n)$. From this we can deduce: $\pi'_1 = A_1^{(q', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ for some q_1, \dots, q_{n+1} . Let $\pi_1 = A_1^{(q, q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$. Let us perform a case analysis on λ :

- *Case:* $\lambda = \epsilon$.

We have $\Pi = \Pi'$, $\Gamma = \Gamma'$, and $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \rightarrow_{\mathcal{P}_0} \pi'_1 \parallel \Pi'_1 \triangleleft \Gamma$ using rule $A_1^{(q, q_1)} \xrightarrow{\epsilon} A_1^{(q', q_1)}$. Further $\pi_1 \in F(\pi)$ and hence $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \in G(\pi \parallel \Pi' \triangleleft \Gamma) = G(\pi \parallel \Pi \triangleleft \Gamma)$. Since $\pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ is an arbitrary element of $G(\pi' \parallel \Pi' \triangleleft \Gamma')$ we can deduce $G(\pi \parallel \Pi \triangleleft \Gamma) \xrightarrow{l_{\mathcal{P}[\mathcal{P}']}} G(\pi' \parallel \Pi' \triangleleft \Gamma')$. and $(\pi' \parallel \Pi' \triangleleft \Gamma', G(\pi' \parallel \Pi' \triangleleft \Gamma')) \in \mathcal{B}$.

- *Case:* $\lambda = c!m$.

We have $\Pi = \Pi'$, $\Gamma' = \Gamma \oplus [c \mapsto [m]]$, and $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \rightarrow_{\mathcal{P}_0} \pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ using rule $A_1^{(q, q_1)} \xrightarrow{c!m} A_1^{(q', q_1)}$. Further $\pi_1 \in F(\pi)$ and hence $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \in G(\pi \parallel \Pi' \triangleleft \Gamma) = G(\pi \parallel \Pi \triangleleft \Gamma)$. Since $\pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ is an arbitrary element of $G(\pi' \parallel \Pi' \triangleleft \Gamma')$ we can deduce $G(\pi \parallel \Pi \triangleleft \Gamma) \xrightarrow{l_{\mathcal{P}[\mathcal{P}']}} G(\pi' \parallel \Pi' \triangleleft \Gamma')$. and $(\pi' \parallel \Pi' \triangleleft \Gamma', G(\pi' \parallel \Pi' \triangleleft \Gamma')) \in \mathcal{B}$.

- *Case:* $\lambda = c?m$.

We have $\Pi = \Pi'$, $\Gamma = \Gamma' \oplus [c \mapsto [m]]$, and $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \rightarrow_{\mathcal{P}_0} \pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ using rule $A_1^{(q, q_1)} \xrightarrow{c?m} A_1^{(q', q_1)}$. Further $\pi_1 \in F(\pi)$ and hence $\pi_1 \parallel \Pi'_1 \triangleleft \Gamma \in G(\pi \parallel \Pi' \triangleleft \Gamma) = G(\pi \parallel \Pi \triangleleft \Gamma)$. Since $\pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ is an arbitrary element of $G(\pi' \parallel \Pi' \triangleleft \Gamma')$ we can deduce $G(\pi \parallel \Pi \triangleleft \Gamma) \xrightarrow{l_{\mathcal{P}[\mathcal{P}']}} G(\pi' \parallel \Pi' \triangleleft \Gamma')$. and $(\pi' \parallel \Pi' \triangleleft \Gamma', G(\pi' \parallel \Pi' \triangleleft \Gamma')) \in \mathcal{B}$.

- *Case:* $\lambda = \nu q'', \epsilon$.

We have $\Pi' = (q'', \epsilon) \parallel \Pi$, $\Gamma = \Gamma'$. Hence $\Pi'_1 = \pi_0 \parallel \Pi_1$ such that $\Pi_1 \triangleleft \Gamma \in G(\Pi \triangleleft \Gamma)$ and $\pi_0 \in F(q'', \epsilon)$ which implies $\pi_0 = \Theta^{q'', q'''}$ for some q''' . Thus

$\pi_1 \parallel \Pi_1 \triangleleft \Gamma \rightarrow_{\mathcal{P}_0} \pi'_1 \parallel \Theta^{q'', q'''} \parallel \Pi_1 \triangleleft \Gamma$ using rule $A_1^{(q, q_1)} \xrightarrow{\nu \Theta^{q'', q'''}} A_1^{(q', q_1)}$. Further $\pi_1 \in F(\pi)$ and hence $\pi_1 \parallel \Pi_1 \triangleleft \Gamma \in G(\pi \parallel \Pi \triangleleft \Gamma)$. Since $\pi'_1 \parallel \Pi'_1 \triangleleft \Gamma'$ is an arbitrary element of $G(\pi' \parallel \Pi' \triangleleft \Gamma')$ we can deduce $G(\pi \parallel \Pi \triangleleft \Gamma) \xrightarrow{l_{\mathcal{P}[\mathcal{P}']}} G(\pi' \parallel \Pi' \triangleleft \Gamma')$. and $(\pi' \parallel \Pi' \triangleleft \Gamma', G(\pi' \parallel \Pi' \triangleleft \Gamma')) \in \mathcal{B}$.

We can thus conclude that \mathcal{B} is a simulation relation.

For a proof that \mathcal{B}^{-1} is a simulation relation we will first prove a little lemma:

Lemma. $\{(s_0, s) : s_0 \in G(s)\}$ is a simulation relation.

Proof. Suppose $\pi_0 \parallel \Pi_0 \triangleleft \Gamma \in G(\pi \parallel \Pi \triangleleft \Gamma)$ and $\pi_0 \parallel \Pi_0 \triangleleft \Gamma \rightarrow_{\mathcal{P}'} \pi'_0 \parallel \Pi'_0 \triangleleft \Gamma'$ using rule $r \in \mathcal{R}'$. Let us make a case analysis on r :

- *Case:* $r = A^{(q, q')} \xrightarrow{\epsilon} \epsilon$.

Then $\Pi'_0 = \Pi_0$, $\Gamma = \Gamma'$, $\pi_0 = A^{(q, q')} A_1^{(q', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ and $\pi'_0 = A_1^{(q', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ for some q_1, \dots, q_{n+1} . We can deduce that $\pi = (q, A A_1 \dots A_n)$ and we can let $\pi' = (q', A_1 \dots A_n)$ so that $\pi'_0 \in F(\pi')$. Further we know by construction that $(q, A) \xrightarrow{\epsilon} (q', \epsilon) \in \mathcal{R}$ and thus $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi' \parallel \Pi \triangleleft \Gamma$ and $\pi'_0 \parallel \Pi'_0 \triangleleft \Gamma' \in G(\pi' \parallel \Pi' \triangleleft \Gamma') = G(\pi' \parallel \Pi \triangleleft \Gamma)$ which is what we wanted to prove.

- *Case:* $r = A^{(q, q'')} \xrightarrow{\epsilon} B^{(q', q''')} A^{(q''', q'')}$.

Then $\Pi'_0 = \Pi_0$, $\Gamma = \Gamma'$, $\pi_0 = A^{(q, q'')} A_1^{(q'', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ and $\pi'_0 = B^{(q', q''')} A^{(q''', q'')} A_1^{(q'', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ for some q_1, \dots, q_{n+1} . We can deduce that $\pi = (q, A A_1 \dots A_n)$ and we can let $\pi' = (q', B A A_1 \dots A_n)$ so that $\pi'_0 \in F(\pi')$. Further we know by construction that $(q, \epsilon) \xrightarrow{\epsilon} (q', B) \in \mathcal{R}$ and thus $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi' \parallel \Pi \triangleleft \Gamma$ and $\pi'_0 \parallel \Pi'_0 \triangleleft \Gamma' \in G(\pi' \parallel \Pi' \triangleleft \Gamma') = G(\pi' \parallel \Pi \triangleleft \Gamma)$ which is what we wanted to prove.

- *Case:* $r = A^{(q, q'')} \xrightarrow{\lambda'} A^{(q', q'')}$.

Then $\Pi'_0 = \Pi_0$, $\Gamma = \Gamma'$, $\pi_0 = A^{(q, q'')} A_1^{(q'', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ and $\pi'_0 = A^{(q', q'')} A_1^{(q'', q_1)} \dots A_n^{(q_n, q_{n+1})} \Theta^{(q_n, q_{n+1})}$ for some q_1, \dots, q_{n+1} . We can deduce that $\pi = (q, A A_1 \dots A_n)$ and we can let $\pi' = (q', A A_1 \dots A_n)$ so that $\pi'_0 \in F(\pi')$. Further we know by construction that $(q, \epsilon) \xrightarrow{\lambda'} (q', \epsilon) \in \mathcal{R}$. Let us perform a case analysis on λ' .

- *Case:* $\lambda' = \epsilon$.

Then $\lambda = \epsilon$, $\Pi = \Pi'$, $\Gamma = \Gamma'$ and thus $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi' \parallel \Pi \triangleleft \Gamma$ and $\pi'_0 \parallel \Pi'_0 \triangleleft \Gamma' \in G(\pi' \parallel \Pi' \triangleleft \Gamma') = G(\pi' \parallel \Pi \triangleleft \Gamma)$ which is what we wanted to prove.

- *Case:* $\lambda' = c!m$.

Then $\lambda = c!m$, $\Pi = \Pi'$, $\Gamma' = \Gamma \oplus [c \mapsto [m]]$ and thus $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi' \parallel \Pi \triangleleft \Gamma'$ and $\pi'_0 \parallel \Pi'_0 \triangleleft \Gamma' \in G(\pi' \parallel \Pi' \triangleleft \Gamma') = G(\pi' \parallel \Pi \triangleleft \Gamma)$ which is what we wanted to prove.

- *Case:* $\lambda' = c ? m$.

Then $\lambda = c ? m$, $\Pi = \Pi'$, $\Gamma = \Gamma' \oplus [c \mapsto [m]]$ and thus $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi' \parallel \Pi \triangleleft \Gamma'$ and $\pi'_0 \parallel \Pi'_0 \triangleleft \Gamma' \in G(\pi' \parallel \Pi' \triangleleft \Gamma') = G(\pi' \parallel \Pi \triangleleft \Gamma')$ which is what we wanted to prove.

- *Case:* $\lambda' = \nu \Theta(q''', q''')$.

Then $\lambda = \nu(q''', \epsilon)$, $\Pi' = (q''', \epsilon) \parallel \Pi$, $\Gamma' = \Gamma$ and thus $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi'(q''', \epsilon) \parallel \Pi \triangleleft \Gamma'$ and $\pi'_0 \parallel \Pi'_0 \triangleleft \Gamma' \in G(\pi' \parallel \Pi' \triangleleft \Gamma') = G(\pi' \parallel \Pi \triangleleft \Gamma')$ which is what we wanted to prove.

which concludes the proof. \square

Now we can use the above Lemma to show that \mathcal{B}^{-1} is a simulation relation. Hence suppose $(\pi \parallel \Pi \triangleleft \Gamma, G(\pi \parallel \Pi \triangleleft \Gamma)) \in \mathcal{B}$ and $G(\pi \parallel \Pi \triangleleft \Gamma) \xrightarrow{L} \mathcal{P} G(\pi' \parallel \Pi' \triangleleft \Gamma')$ then clearly $G(\pi' \parallel \Pi' \triangleleft \Gamma') \neq \emptyset$ and hence we have $s \in G(\pi \parallel \Pi \triangleleft \Gamma)$ and $s' \in G(\pi' \parallel \Pi' \triangleleft \Gamma')$ such that $s \rightarrow_{\mathcal{P}'} s'$ from which the above Lemma let's us conclude that $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi' \parallel \Pi' \triangleleft \Gamma'$. Hence we can conclude that \mathcal{B} is a bisimulation.

Let us define $\mathcal{F}^{\text{nf}}(\mathcal{P}) = (\mathcal{Q}', \mathcal{A}' \cup \mathcal{A}_{\text{cov}}, \text{Chan}, \text{Msg}, \mathcal{R}' \cup \mathcal{R}_{\text{cov}})$ where $\mathcal{A}_{\text{cov}} = \{\Theta_{\text{cov}}^{(q,A)}, \Theta_{\text{cov}}^{(q,\epsilon)} : q \in \mathcal{Q}, A \in \mathcal{A}\}$ and

$$\mathcal{R}_{\text{cov}} = \left\{ A^{q,q'} \xrightarrow{\epsilon} \Theta_{\text{cov}}^{(q,A)}, A^{q,q'} \xrightarrow{\epsilon} \Theta_{\text{cov}}^{(q,\epsilon)} : q, q' \in \mathcal{Q}, A \in \mathcal{A} \right\} \cup \left\{ \Theta_{\text{cov}}^{q,q'} \xrightarrow{\epsilon} \Theta_{\text{cov}}^{(q,\epsilon)} : q, q' \in \mathcal{Q} \right\}.$$

It is easy to see that $\mathcal{F}^{\text{nf}}(\mathcal{P})$ is polynomial-time computable from \mathcal{P} .

Now suppose $Q = (\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}})$ is a simple coverability query. Since Q is simple we may assume that $\Pi_0 = (q_1^0, \epsilon) \parallel \dots \parallel (q_k^0, \epsilon)$ and $\Pi_{\text{cov}} = (q_1, \beta_1) \parallel \dots \parallel (q_n, \beta_n)$ such that $\beta_i \in \mathcal{A} \cup \{\epsilon\}$ since a trivial polynomial-time transform \mathcal{P} may set up a general $\Pi_0 \triangleleft \Gamma_0$ from a simple query Q .

Fix a $q^\dagger \in \mathcal{Q}$. We may define the query $\mathcal{F}^{\text{nf}}(Q) = (\mathcal{F}^{\text{nf}}(\mathcal{P}), \mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0), \mathcal{F}^{\text{nf}}(\Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}}))$ where

$$\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) = \Theta_{\text{cov}}^{(q_1^0, q^\dagger)} \parallel \dots \parallel \Theta_{\text{cov}}^{(q_k^0, q^\dagger)} \triangleleft \Gamma_0 \text{ and}$$

$$\mathcal{F}^{\text{nf}}(\Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}}) = \Theta_{\text{cov}}^{(q_1, \beta_1)} \parallel \dots \parallel \Theta_{\text{cov}}^{(q_n, \beta_n)} \triangleleft \Gamma.$$

$\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \in G(\Pi_0 \triangleleft \Gamma_0)$ and $\mathcal{F}^{\text{nf}}(\Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}}) \in G(\Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}})$ and both the former are polynomial-time computable.

Now suppose Q is a yes-instance, then $\Pi_0 \triangleleft \Gamma_0 \rightarrow_{\mathcal{P}}^* \Pi \triangleleft \Gamma$ such that $\Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}} \leq_{\text{ACPS}} \Pi \triangleleft \Gamma$ and since \mathcal{B} is a bisimulation we also know that $G(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{P}'}^* G(\Pi \triangleleft \Gamma)$. Further we must have $\Pi = (q_1, \beta'_1) \parallel \dots \parallel (q_n, \beta'_n) \parallel \Pi_1$ where either $\beta_i = \epsilon$ or $\beta_i = A_i$ and $\beta'_i = A_i \cdot \beta''_i$. By definition for all $\Pi' \triangleleft \Gamma \in G(\Pi \triangleleft \Gamma)$ there exists a $\Pi'_0 \triangleleft \Gamma_0 \in G(\Pi_0 \triangleleft \Gamma_0)$ such that $\Pi'_0 \triangleleft \Gamma_0 \rightarrow_{\mathcal{P}'}^* \Pi' \triangleleft \Gamma$. We can deduce that $\Pi' = A_1^{q_1, q'_1} \bar{\beta}_1 \parallel \dots \parallel A_n^{q_n, q'_n} \bar{\beta}_n \parallel \Pi'_1$ where either $A_i = \beta_i$ or $\beta_i = \epsilon$ and $A_i = \Theta$. We can then see that $\Pi' \triangleleft \Gamma \rightarrow_{\mathcal{F}^{\text{nf}}(\mathcal{P})}^* \Theta_{\text{cov}}^{(q_1, \beta_1)} \bar{\beta}_1 \parallel \dots \parallel \Theta_{\text{cov}}^{(q_n, \beta_n)} \bar{\beta}_n \parallel \Pi'_1$. Since $\Pi'_0 = \Theta(q_1^0, q_1^1) \parallel \dots \parallel \Theta(q_k^0, q_k^1)$ may differ from $\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0)$ only by the choice of the q_j^1 which cannot play a rôle in any

reduction; this allows us to deduce: $\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{F}^{\text{nf}}(\mathcal{P})}^* \Theta_{\text{cov}}^{(q_1, \beta_1)} \bar{\beta}_1 \parallel \dots \parallel \Theta_{\text{cov}}^{(q_n, \beta_n)} \bar{\beta}_n \parallel \Pi'_1$. Thus $\mathcal{F}^{\text{nf}}(Q)$ is a yes-instance.

Conversely, suppose $\mathcal{F}^{\text{nf}}(Q)$ is a yes-instance then $\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{F}^{\text{nf}}(\mathcal{P})}^* \Theta_{\text{cov}}^{(q_1, \beta_1)} \bar{\beta}_1 \parallel \dots \parallel \Theta_{\text{cov}}^{(q_n, \beta_n)} \bar{\beta}_n \parallel \Pi'_1$. Reversing transitions using rules from \mathcal{R}_{cov} we can see that $\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{P}'}^* A_1^{q_1, q'_1} \bar{\beta}_1 \parallel \dots \parallel A_n^{q_n, q'_n} \bar{\beta}_n \parallel \Pi'_1$ where either $A_i = \beta_i$ or $\beta_i = \epsilon$ and $A_i = \Theta$. Since $\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \in G(\Pi_0 \triangleleft \Gamma_0)$ the Lemma above implies that $A_1^{q_1, q'_1} \bar{\beta}_1 \parallel \dots \parallel A_n^{q_n, q'_n} \bar{\beta}_n \parallel \Pi'_1 \in G((q_1, \beta_1 \beta'_1) \parallel \dots \parallel (q_n, \beta_n \beta'_n) \parallel \Pi_1)$ for some β'_i and Π_1 and $\Pi_0 \triangleleft \Gamma_0 \rightarrow_{\mathcal{P}}^* (q_1, \beta_1 \beta'_1) \parallel \dots \parallel (q_n, \beta_n \beta'_n) \parallel \Pi_1$. Thus $(\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi_{\text{cov}} \triangleleft \Gamma_{\text{cov}})$ is a yes-instance.

For boundedness, suppose given $\Pi_0 \triangleleft \Gamma_0$ suppose $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \rightarrow_{\mathcal{P}}^* \Pi \triangleleft \Gamma\}$ is a finite set. Then since \mathcal{B} is a bisimulation this implies $\{G(\Pi \triangleleft \Gamma) : G(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{P}'}^* G(\Pi \triangleleft \Gamma)\}$ is a finite set which implies that $\{\Pi \triangleleft \Gamma : \mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{P}'}^* \Pi \triangleleft \Gamma\}$ is a finite set since $G(\Pi \triangleleft \Gamma)$ is a finite set for all $\Pi \triangleleft \Gamma$. Since rules from \mathcal{R}_{cov} only add a finite of finite number of configurations we can deduce that $\{\Pi \triangleleft \Gamma : \mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{F}^{\text{nf}}(\mathcal{P})}^* \Pi \triangleleft \Gamma\}$ is a finite set. Conversely, suppose $\{\Pi \triangleleft \Gamma : \mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{F}^{\text{nf}}(\mathcal{P})}^* \Pi \triangleleft \Gamma\}$ is a finite set then since rules from \mathcal{R}_{cov} only add a finite of finite number of configurations we can infer that $\{\Pi \triangleleft \Gamma : \mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{P}'}^* \Pi \triangleleft \Gamma\}$ is finite set. Clearly this implies that $\{G(\Pi \triangleleft \Gamma) : G(\Pi_0 \triangleleft \Gamma_0) \rightarrow_{\mathcal{P}'}^* G(\Pi \triangleleft \Gamma)\}$ is a finite set and thus $\{\Pi \triangleleft \Gamma : \Pi_0 \triangleleft \Gamma_0 \rightarrow_{\mathcal{P}}^* \Pi \triangleleft \Gamma\}$ is a finite set since \mathcal{B} is a bisimulation.

For termination, since \mathcal{B} is a bisimulation clearly there is an infinite path from $\Pi_0 \triangleleft \Gamma_0$ in \mathcal{P} iff there is an infinite path from $G(\Pi_0 \triangleleft \Gamma_0)$ in \mathcal{P}' . And the latter is clearly possible if, and only if, there is an infinite path from a $\mathcal{F}^{\text{nf}}(\Pi_0 \triangleleft \Gamma_0) \in G(\Pi_0 \triangleleft \Gamma_0)$ which is implied by the Lemma above and by \mathcal{B} being a bisimulation. This concludes the proof. \square

We can now give a proof of Proposition 1:

Proof of Proposition 1. Let $\mathcal{F}'(\mathcal{P}) = \mathcal{F}^{\text{nf}}(\mathcal{F}^{\text{pnf}}(\mathcal{P}))$, $\mathcal{F}'(Q) = \mathcal{F}^{\text{nf}}(\mathcal{F}^{\text{pnf}}(Q))$, and $\mathcal{F}'(\Pi_0 \triangleleft \Gamma_0) = \mathcal{F}^{\text{nf}}(\mathcal{F}^{\text{pnf}}(\Pi_0 \triangleleft \Gamma_0))$. Lemma 7 and Lemma 8 then implies that $\mathcal{F}'(Q)$ is a yes-instance if, and only if, Q is a yes-instance; $\mathcal{F}'(\mathcal{P})$ is bounded from $\mathcal{F}'(\Pi_0 \triangleleft \Gamma_0)$ if, and only if, \mathcal{P} is bounded from $\Pi_0 \triangleleft \Gamma_0$; and $\mathcal{F}'(\mathcal{P})$ is terminating from $\mathcal{F}'(\Pi_0 \triangleleft \Gamma_0)$ if, and only if, \mathcal{P} is terminating from $\Pi_0 \triangleleft \Gamma_0$. $\mathcal{F}'(\mathcal{P})$ is not quite in normal form yet, since it contains rules of the form $A \xrightarrow{\lambda} B$ where normal form requires the RHS to be ϵ , clearly we can by introducing a polynomial number of non-terminals remedy this; replacing such rules by pairs::

$$A \xrightarrow{\lambda} B \mapsto \left\{ A \xrightarrow{\epsilon} C^\lambda B, C^\lambda \xrightarrow{\lambda} \epsilon \right\}.$$

We call the the resulting ACPS $\mathcal{F}(\mathcal{P})$, and take the query $\mathcal{F}(Q) = \mathcal{F}'(Q)$, and $\mathcal{F}(\Pi_0 \triangleleft \Gamma_0) = \mathcal{F}'(\Pi_0 \triangleleft \Gamma_0)$ and it is

trivial to see that the result holds. \square

2) *Proof of Proposition 2:* We fix a \mathcal{P} in normal form for this section and hence have also a fixed $\mathcal{G}(\mathcal{P})$. The proof of Proposition 2 exploits the fact, that we may accelerate any transitions with commutative side-effects (send, spawn, ϵ) and that we may delay blocking/non-commutative transitions until the last possible point. This means we can synchronise reductions of \mathcal{P} and $\mathcal{G}(\mathcal{P})$ at configurations where all processes have non-commutative non-terminals as a head symbol. We will relabel the transition relations several times in order to chose different synchronisation points.

Let us first extend $\simeq_{I(\mathcal{G}(\mathcal{P}))}$ to parallel compositions. We write $\pi_1 \parallel \dots \parallel \pi_n \simeq_{I(\mathcal{G}(\mathcal{P}))} \bar{\pi}_1 \parallel \dots \parallel \bar{\pi}_n$ just if $\pi_i \simeq_{I(\mathcal{G}(\mathcal{P}))} \bar{\pi}_i$ for all $i \in \langle n \rangle$.

For this section let us write $\rightarrow_{\mathcal{G}(\mathcal{P})}$ for \rightarrow_{con} to make clear that the transition relation is induced by $\mathcal{G}(\mathcal{P})$. We temporarily label $\rightarrow_{\mathcal{P}}$ and $\rightarrow_{\mathcal{G}(\mathcal{P})}$ by \mathcal{P} -configurations as follows: If $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \Pi' \triangleleft \Gamma'$ then we label the transition $\Pi \triangleleft \Gamma \xrightarrow{\Pi_0 \triangleleft \Gamma'} \rightarrow_{\mathcal{P}} \Pi' \triangleleft \Gamma'$ for $\Pi_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi'$. If $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$ and $\Pi' \simeq_{I(\mathcal{G}(\mathcal{P}))} \bar{\Pi}'$, $\bar{\Pi}' \in \mathbb{M}[\mathcal{N}^*]$ then we label the transition $\Pi \triangleleft \Gamma \xrightarrow{\bar{\Pi}' \triangleleft \Gamma'} \rightarrow_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$; otherwise we label it by ϵ , i.e. $\Pi \triangleleft \Gamma \xrightarrow{\epsilon} \rightarrow_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$.

Lemma 9. The relation $\mathcal{R} = \{(\Pi \triangleleft \Gamma, \bar{\Pi} \triangleleft \Gamma') : \bar{\Pi} \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi, \bar{\Pi} \in \mathbb{M}[\mathcal{N}^*]\}$ is a weak simulation relation for \mathcal{P} and $\mathcal{G}(\mathcal{P})$.

Proof. Suppose $(\pi \parallel \Pi \triangleleft \Gamma, \pi_0 \parallel \Pi_0 \triangleleft \Gamma) \in \mathcal{R}$ and $\pi \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \pi' \parallel \Pi' \triangleleft \Gamma'$. We may assume that $\pi = A\beta$ and a rule $A \xrightarrow{\lambda} \beta'$ is used in the transition. Since $\pi_0 \parallel \Pi_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} A\beta \parallel \Pi$ we can w.l.o.g. assume that $\pi_0 = A\beta$ and $\beta \simeq_{I(\mathcal{G}(\mathcal{P}))} \bar{\beta}$ (by using transitivity of $\simeq_{I(\mathcal{G}(\mathcal{P}))}$ otherwise). Let us perform a case analysis on λ :

- Case: $\lambda = \epsilon$

Since \mathcal{P} is in normal form we know that $\beta' \in \{\epsilon, BC : B, C \in \mathcal{A}\}$, $\pi' = \beta' \beta$, $\Pi' = \Pi$, $\Gamma' = \Gamma$ and $A \rightarrow \beta' \in \mathcal{G}(\mathcal{P})$. Then clearly $A\bar{\beta} \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l} \rightarrow_{\mathcal{G}(\mathcal{P})} \beta' \bar{\beta} \parallel \Pi_0 \triangleleft \Gamma$ and $\beta' \bar{\beta} \parallel \Pi_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta' \beta \parallel \Pi = \pi' \parallel \Pi'$. Further clearly $\beta' \bar{\beta} \parallel \Pi_0 \in \mathbb{M}[\mathcal{N}^*]$ and thus we may take $l = \pi' \parallel \Pi'$.

- Case: $\lambda = c!m$

Since \mathcal{P} is in normal form we know that $\beta' = \epsilon$, $\pi' = \beta$, $\Pi' = \Pi$, $\Gamma' = \Gamma \oplus [c \mapsto [m]]$ and $A \rightarrow c!m \in \mathcal{G}(\mathcal{P})$. Then clearly $A\bar{\beta} \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{\epsilon} \rightarrow_{\mathcal{G}(\mathcal{P})} c!m\bar{\beta} \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l} \rightarrow_{\mathcal{G}(\mathcal{P})} \bar{\beta} \parallel \Pi_0 \triangleleft \Gamma'$ and $\bar{\beta} \parallel \Pi_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta \parallel \Pi = \pi' \parallel \Pi'$. Further clearly $\bar{\beta} \parallel \Pi_0 \in \mathbb{M}[\mathcal{N}^*]$ and thus we may take $l = \pi' \parallel \Pi'$.

- Case: $\lambda = c?m$

Since \mathcal{P} is in normal form we know that $\beta' = \epsilon$, $\pi' = \beta$, $\Pi' = \Pi$, $\Gamma' = \Gamma \oplus [c \mapsto [m]]$ and $A \rightarrow c?m \in \mathcal{G}(\mathcal{P})$. Then clearly $A\bar{\beta} \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{\epsilon} \rightarrow_{\mathcal{G}(\mathcal{P})} c?m\bar{\beta} \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l} \rightarrow_{\mathcal{G}(\mathcal{P})} \bar{\beta} \parallel \Pi_0 \triangleleft \Gamma'$ and $\bar{\beta} \parallel \Pi_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta \parallel \Pi = \pi' \parallel \Pi'$. Further clearly $\bar{\beta} \parallel \Pi_0 \in \mathbb{M}[\mathcal{N}^*]$ and thus we may take $l = \pi' \parallel \Pi'$.

- Case: $\lambda = \nu A'$

Since \mathcal{P} is in normal form we know that $\beta' = \epsilon$, $\pi' = \beta$,

$\Pi' = A' \parallel \Pi$, $\Gamma = \Gamma'$ and $A \rightarrow \nu A' \in \mathcal{G}(\mathcal{P})$. Then clearly $A\bar{\beta} \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{\epsilon} \rightarrow_{\mathcal{G}(\mathcal{P})} \nu A' \bar{\beta} \parallel \Pi_0 \triangleleft \Gamma \xrightarrow{l} \rightarrow_{\mathcal{G}(\mathcal{P})} \bar{\beta} \parallel A' \parallel \Pi_0 \triangleleft \Gamma'$. Further $\bar{\beta} \parallel A' \parallel \Pi_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta \parallel A' \parallel \Pi = \pi' \parallel \Pi'$, and Further clearly $\bar{\beta} \parallel A' \parallel \Pi_0 \in \mathbb{M}[\mathcal{N}^*]$ and thus we may take $l = \pi' \parallel \Pi'$.

Hence \mathcal{R} is a weak simulation which is what we wanted to prove. \square

We temporarily relabel $\rightarrow_{\mathcal{P}}$ and $\rightarrow_{\mathcal{G}(\mathcal{P})}$ by side effects in Λ as follows: If $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \Pi' \triangleleft \Gamma'$ using rule $\beta \xrightarrow{\lambda} \beta'$ then we label the transition $\Pi \triangleleft \Gamma \xrightarrow{\lambda} \rightarrow_{\mathcal{P}} \Pi' \triangleleft \Gamma'$. If $\lambda \beta \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}(\mathcal{P})} \beta \parallel \Pi' \triangleleft \Gamma'$ using rules (R-4), (R-5) and $\lambda \in \Sigma(\mathcal{P})$ then we label the transition $\lambda \beta \parallel \Pi \triangleleft \Gamma \xrightarrow{\lambda} \rightarrow_{\mathcal{G}(\mathcal{P})} \beta \parallel \Pi' \triangleleft \Gamma'$; otherwise we label it by ϵ , i.e. $\Pi \triangleleft \Gamma \xrightarrow{\epsilon} \rightarrow_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$. Further we label the transitive closures as usual: If $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{l_1} \rightarrow_{\mathcal{G}(\mathcal{P})} \Pi_1 \triangleleft \Gamma_1 \xrightarrow{l_2} \rightarrow_{\mathcal{G}(\mathcal{P})} \dots \xrightarrow{l_n} \rightarrow_{\mathcal{G}(\mathcal{P})} \Pi_n \triangleleft \Gamma_n$ then we label $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{l_1 \dots l_n} \rightarrow_{\mathcal{G}(\mathcal{P})}^* \Pi_n \triangleleft \Gamma_n$ and similarly for $\xrightarrow{l} \rightarrow_{\mathcal{P}}^*$.

We capture the language of side effects of a $\beta \in \mathcal{N}^{\text{com}}$ in \mathcal{P} and $\mathcal{G}(\mathcal{P})$ as follows: $\mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta) = \{\lambda \mid \beta \triangleleft \emptyset \xrightarrow{\lambda} \rightarrow_{\mathcal{G}(\mathcal{P})}^* \epsilon \parallel \Pi(\lambda) \triangleleft \Gamma(\lambda)\}$ and $\mathcal{L}_{\mathcal{P}}(\beta) = \{\lambda \mid \beta \triangleleft \emptyset \xrightarrow{\lambda} \rightarrow_{\mathcal{P}}^* \epsilon \parallel \Pi(\lambda) \triangleleft \Gamma(\lambda)\}$ where a λ determines a “delta” to the configuration in terms of sent messages and spawned processes:

$$\Pi(\lambda) := \left\{ \prod_{A \in \mathcal{N}} \left(\prod_1^{\mathbb{M}(\lambda)(\nu A)} A \right) \right\},$$

$$\Gamma(\lambda) := \left\{ \bigoplus_{c \in \text{Chan}} \bigoplus_{m \in \text{Msg}} \left[c \mapsto \left[m^{\mathbb{M}(\lambda)(c!m)} \right] \right] \right\}.$$

Further we note that $\mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta)$ is effectively the Parikh language defined by the CFG $\mathcal{G}(\mathcal{P})$ of β in the standard derivation sense. We can now make precise how we can accelerate the execution of commutative non-terminals:

Lemma 10. Suppose $\beta, \bar{\beta} \in \mathcal{N}^{\text{com}*}$ then:

- (i) $\mathcal{L}_{\mathcal{P}}(\beta) \neq \emptyset$ and for all $\lambda \in \mathcal{L}_{\mathcal{P}}(\beta)$ we have the transitions: $\beta \beta' \parallel \Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \beta' \parallel \Pi \parallel \Pi(\lambda) \triangleleft \Gamma \oplus \Gamma(\lambda)$; and
- (ii) $\mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta}) \neq \emptyset$ and for all $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta})$ we have the transitions: $\beta \bar{\beta}' \parallel \bar{\Pi} \triangleleft \bar{\Gamma} \rightarrow_{\mathcal{G}(\mathcal{P})}^* \bar{\beta}' \parallel \bar{\Pi} \parallel \Pi(\lambda) \triangleleft \Gamma(\lambda)$.

And if $\beta \simeq_{I(\mathcal{G}(\mathcal{P}))} \bar{\beta}$ then we have the set equality:

$$\{(\Pi(\lambda), \Gamma(\lambda)) \mid \lambda \in \mathcal{L}_{\mathcal{P}}(\beta)\} = \{(\Pi(\lambda), \Gamma(\lambda)) \mid \lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta})\}.$$

Proof. Let us first prove that $\mathcal{L}_{\mathcal{P}}(\beta) \neq \emptyset$. First since $\beta \in \mathcal{N}^{\text{com}*}$ we may see them as non-terminals of $\mathcal{G}(\mathcal{P})$ and rewrite them. Suppose $\beta = A_1 \dots A_n$ then we know that A_i is commutative and thus $\mathcal{L}(A_i) \neq \emptyset$ for all $i \in \langle n \rangle$. So let us pick $\lambda_i \in \mathcal{L}(A_i)$ let us show in a brief induction that if $\lambda \in \mathcal{L}(A)$ then $\lambda \in \mathcal{L}_{\mathcal{P}}(A)$. Let us suppose $A \rightarrow^n \lambda$. If $n = 1$ then we use a rule $A \rightarrow \lambda$ in $\mathcal{G}(\mathcal{P})$; hence there exists a rule $A \xrightarrow{\lambda} \epsilon$ in \mathcal{P} and thus $A \triangleleft \emptyset \xrightarrow{\lambda} \rightarrow_{\mathcal{G}(\mathcal{P})}^* \epsilon \parallel \Pi(\lambda) \triangleleft \Gamma(\lambda)$. If $n = k + 1$ and the claim holds for all A' and $k' \leq k$ then we can see that $A \rightarrow \beta \rightarrow^k \lambda$. Now since $\mathcal{G}(\mathcal{P})$ is in Chomsky normal form the first step must be a rule of the form $A \rightarrow BC$ and $\beta = BC$. This implies $\lambda = \lambda' \lambda''$, $B \rightarrow^{k'} \lambda'$, $C \rightarrow^{k''} \lambda''$, and $k = k' + k''$,

which yields using the IH that $\lambda' \in \mathcal{L}_{\mathcal{P}}(B)$ and $\lambda'' \in \mathcal{L}_{\mathcal{P}}(C)$. Further we know there is a rule $A \xrightarrow{\epsilon} BC$ in \mathcal{P} and thus $A \triangleleft \emptyset \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})}^* BC \triangleleft \emptyset \xrightarrow{\lambda'}_{\mathcal{G}(\mathcal{P})}^* C \parallel \Pi(\lambda') \triangleleft \Gamma(\lambda') \xrightarrow{\lambda''}_{\mathcal{G}(\mathcal{P})}^* \epsilon \parallel \Pi(\lambda' \lambda'') \triangleleft \Gamma(\lambda'')$ which concludes the induction. Hence we can infer that in fact $\lambda_i \in \mathcal{L}(A_i)$ and so we can see that $A_1 \cdots A_n \triangleleft \emptyset \xrightarrow{\lambda_1}_{\mathcal{G}(\mathcal{P})}^* A_2 \cdots A_n \parallel \Pi(\lambda_1) \triangleleft \Gamma(\lambda_1) \xrightarrow{\lambda_2}_{\mathcal{G}(\mathcal{P})}^* \cdots \xrightarrow{\lambda_n}_{\mathcal{G}(\mathcal{P})}^* \cdots \epsilon \parallel \Pi(\lambda_1 \cdots \lambda_n) \triangleleft \Gamma(\lambda_1 \cdots \lambda_n)$ and thus $\lambda_1 \cdots \lambda_n \in \mathcal{L}_{\mathcal{P}}(\beta)$ which is what we wanted to prove. For the second claim of (i) suppose $\lambda \in \mathcal{L}_{\mathcal{P}}(\beta)$ then by definition $\beta \triangleleft \emptyset \xrightarrow{\lambda}_{\mathcal{G}(\mathcal{P})}^* \epsilon \parallel \Pi(\lambda) \triangleleft \Gamma(\lambda)$. It is then trivial to see that $\beta \beta' \parallel \Pi \triangleleft \Gamma \xrightarrow{\beta'}_{\mathcal{P}} \beta' \parallel \Pi \parallel \Pi(\lambda) \triangleleft \Gamma \oplus \Gamma(\lambda)$.

For (ii), let us first prove that $\mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta}) \neq \emptyset$. First we note again that $\bar{\beta} \in \mathcal{N}^{\text{com}*}$ and thus $\mathcal{L}(\bar{\beta}) \neq \emptyset$. Hence take $\lambda \in \mathcal{L}(\bar{\beta})$. By continuously commuting non-terminals to the leftmost-position we can see that $\bar{\beta} \triangleleft \emptyset \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})}^* \lambda \triangleleft \emptyset$ and then clearly $\lambda \triangleleft \emptyset \xrightarrow{\lambda}_{\mathcal{G}(\mathcal{P})}^* \epsilon \parallel \Pi(\lambda) \triangleleft \Gamma(\lambda)$ thus $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta})$. For the second claim of (ii) let $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta})$ then by definition $\bar{\beta} \triangleleft \emptyset \xrightarrow{\lambda}_{\mathcal{G}(\mathcal{P})}^* \epsilon \parallel \Pi(\lambda) \triangleleft \Gamma(\lambda)$; it is hence trivial to see that $\bar{\beta} \beta' \parallel \bar{\Pi} \triangleleft \bar{\Gamma} \xrightarrow{\beta'}_{\mathcal{P}} \beta' \parallel \bar{\Pi} \parallel \Pi(\lambda) \triangleleft \bar{\Gamma} \oplus \Gamma(\lambda)$.

Since $\mathcal{G}(\mathcal{P})$, restricted to non-terminals of \mathcal{N}^{com} , is effectively a completely commutative context-free grammar derived from \mathcal{P} we can see that $\{\mathbb{M}(\lambda) \mid \lambda \in \mathcal{L}_{\mathcal{P}}(\beta)\} = \{\mathbb{M}(\bar{\lambda}) \mid \bar{\lambda} \simeq_{I(\mathcal{G}(\mathcal{P}))} \lambda \in \mathcal{L}_{\mathcal{P}}(\beta)\} = \{\mathbb{M}(\bar{\lambda}) \mid \bar{\lambda} \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta})\}$ which implies immediately that

$$\{(\Pi(\lambda), \Gamma(\lambda)) \mid \lambda \in \mathcal{L}_{\mathcal{P}}(\beta)\} = \{(\Pi(\lambda), \Gamma(\lambda)) \mid \lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\bar{\beta})\}.$$

□

As a final step in our argument let us relabel the transition relation again. We temporarily relabel $\rightarrow_{\mathcal{P}}$ and $\rightarrow_{\mathcal{G}(\mathcal{P})}$ by \mathcal{P} -configurations as follows: If $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{P}} \Pi' \triangleleft \Gamma'$ then we label the transition $\Pi \triangleleft \Gamma \xrightarrow{\Pi_0 \triangleleft \Gamma'}_{\mathcal{P}} \Pi' \triangleleft \Gamma'$, if $\Pi' \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{A}^*]$ and $\Pi'_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi'$; otherwise we label it with ϵ : $\Pi \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}} \Pi' \triangleleft \Gamma'$. If $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$ and $\Pi' \simeq_{I(\mathcal{G}(\mathcal{P}))} \bar{\Pi}'$, $\bar{\Pi}' \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{N}^*]$ then we label the transition $\Pi \triangleleft \Gamma \xrightarrow{\bar{\Pi}' \triangleleft \Gamma'}_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$; otherwise we label it by ϵ , i.e. $\Pi \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$.

Lemma 11. The relation $\mathcal{R}' = \{(\Pi \triangleleft \Gamma, \bar{\Pi} \triangleleft \bar{\Gamma}) : \bar{\Pi} \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi, \bar{\Pi} \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{N}^*]\}$ is a weak bisimulation relation for \mathcal{P} and $\mathcal{G}(\mathcal{P})$.

Proof. Let us first prove that \mathcal{R}' is a weak simulation. Suppose $(\Pi \triangleleft \Gamma, \Pi_0 \triangleleft \Gamma) \in \mathcal{R}'$ and $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{P}}^* \Pi' \triangleleft \Gamma'$ such that $\Pi' \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{A}^*]$. Lemma 9 then tells us that $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{P}}^* \Pi'_0 \triangleleft \Gamma'$ and $\Pi'_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi'$. The latter implies that $\Pi'_0 \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{N}^*]$ since $\Pi' \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{A}^*]$, i.e. each process is headed by a non-commutative non-terminal which obviously are invariant under commutation. Hence we can deduce that \mathcal{R}' is a weak simulation.

Let us first prove that \mathcal{R}'^{-1} is a weak simulation. Suppose $(\Pi \triangleleft \Gamma, \Pi_0 \triangleleft \Gamma) \in \mathcal{R}'$ and $\Pi_0 \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}(\mathcal{P})}^* \Pi'_0 \triangleleft \Gamma'$ where $\Pi'_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi'$ for some $\Pi' \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{A}^*]$. W.l.o.g.

we may assume this path may not be split to expose two labels l, l' (otherwise we may consider a maximal splitting of the path to obtain several labels and apply the below to each in turn one-by-one). We can infer that $\Pi_0 = A \beta_0 \parallel \Pi_1$ and the transition may be split $A \beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{l}_{\mathcal{G}(\mathcal{P})} \beta' \beta_0 \parallel \Pi'_1 \triangleleft \Gamma_1 \xrightarrow{l'}_{\mathcal{G}(\mathcal{P})}^* \Pi'_0 \triangleleft \Gamma'$, $l \cdot l' = \Pi' \triangleleft \Gamma'$ and using a rule $A \rightarrow \beta' \in \mathcal{G}(\mathcal{P})$. Further we may infer that $\Pi = A \beta \parallel \Pi_2$. Let us make a case analysis on β' .

- *Case:* $\beta' = BC$, B non-commutative.

Firstly, we notice $\Pi'_1 = \Pi_1$, $\Gamma = \Gamma_1$ and $BC \beta_0 \parallel \Pi_1 \in \mathbb{M}[\mathcal{N}^{\text{com}} \cdot \mathcal{N}^*]$ which implies that $l = \Pi' \triangleleft \Gamma'$ and also $\Pi'_0 \triangleleft \Gamma' = \beta' \beta_0 \parallel \Pi_1$. Further there is a rule $A \xrightarrow{\epsilon} BC$ in \mathcal{P} and thus $A \beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{l}_{\mathcal{P}} BC \beta \parallel \Pi_2 \triangleleft \Gamma$ and $BC \beta \parallel \Pi_2 \simeq_{I(\mathcal{G}(\mathcal{P}))} BC \beta_0 \parallel \Pi_1$ which is what we wanted to prove.

- *Case:* $\beta' = BC$, B commutative.

Firstly, we notice $\Pi'_1 = \Pi_1$ and $\Gamma = \Gamma_1$. Since A was non-commutative it must be the case that C is non-commutative. Further since $\Pi_0 \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}(\mathcal{P})}^* \Pi'_0 \triangleleft \Gamma'$ may not be further split we may assume that for some $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(B)$ we may rewrite the transition as: $A \beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} BC \beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{l'}_{\mathcal{G}(\mathcal{P})}^* C \beta_0 \parallel \Pi_1 \parallel \Pi(\lambda) \triangleleft \Gamma \oplus \Gamma(\lambda)$, i.e. $\Pi'_0 \triangleleft \Gamma' = C \beta_0 \parallel \Pi_1 \parallel \Pi(\lambda) \triangleleft \Gamma \oplus \Gamma(\lambda)$. Now we know that there is a rule $A \xrightarrow{\epsilon} BC$ in \mathcal{P} and thus $A \beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}} BC \beta \parallel \Pi_2 \triangleleft \Gamma$ and Lemma 10 then gives us that $BC \beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{l'}_{\mathcal{P}}^* C \beta \parallel \Pi_2 \parallel \Pi(\bar{\lambda}) \triangleleft \Gamma \oplus \Gamma(\bar{\lambda})$ for some $\bar{\lambda} \in \mathcal{L}_{\mathcal{P}}(B)$ such that $\Pi(\bar{\lambda}) = \Pi(\lambda)$, $\Gamma(\bar{\lambda}) = \Gamma(\lambda)$. Further $C \beta \parallel \Pi_2 \parallel \Pi(\lambda) \simeq_{I(\mathcal{G}(\mathcal{P}))} C \beta_0 \parallel \Pi_1 \parallel \Pi(\lambda)$ which is what we wanted to prove.

- *Case:* $\beta' = \epsilon$.

Firstly, we notice $\Pi'_1 = \Pi_1$ and $\Gamma = \Gamma_1$ and we may write $\beta_0 = \beta'_0 \beta''_0$ such that $\beta'_0 \in \mathcal{N}^{\text{com}*}$ and $\beta''_0 \in (\mathcal{N}^{\text{com}} \mathcal{N}^{\text{com}*})^*$. Further since $\Pi_0 \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}(\mathcal{P})}^* \Pi'_0 \triangleleft \Gamma'$ may not be further split we may assume that for some $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta'_0)$ we may rewrite the transition as: $A \beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{l'}_{\mathcal{G}(\mathcal{P})}^* \beta''_0 \parallel \Pi_1 \parallel \Pi(\lambda) \triangleleft \Gamma \oplus \Gamma(\lambda)$, i.e. $\Pi'_0 \triangleleft \Gamma' = \beta''_0 \parallel \Pi_1 \parallel \Pi(\lambda) \triangleleft \Gamma \oplus \Gamma(\lambda)$. Now we know that there is a rule $A \xrightarrow{\epsilon} \epsilon$ in \mathcal{P} and thus $A \beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}} \beta \parallel \Pi_2 \triangleleft \Gamma$ and since $\beta \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta_0$ we may rewrite $\beta = \beta' \beta''$ such that $\beta' \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta'_0$ and $\beta'' \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta''_0$, hence clearly $\beta' \in \mathcal{N}^{\text{com}*}$ and $\beta'' \in (\mathcal{N}^{\text{com}} \mathcal{N}^{\text{com}*})^*$, and Lemma 10 then gives us that $\beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{l'}_{\mathcal{P}}^* \beta'' \parallel \Pi_2 \parallel \Pi(\bar{\lambda}) \triangleleft \Gamma \oplus \Gamma(\bar{\lambda})$ for some $\bar{\lambda} \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta')$ such that $\Pi(\bar{\lambda}) = \Pi(\lambda)$, $\Gamma(\bar{\lambda}) = \Gamma(\lambda)$. Further $\beta'' \parallel \Pi_2 \parallel \Pi(\lambda) \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta''_0 \parallel \Pi_1 \parallel \Pi(\lambda)$ which is what we wanted to prove.

- *Case:* $\beta' = c ! m$.

Firstly, we notice $\Pi'_1 = \Pi_1$ and $\Gamma = \Gamma_1$ and we may write $\beta_0 = \beta'_0 \beta''_0$ such that $\beta'_0 \in \mathcal{N}^{\text{com}*}$ and $\beta''_0 \in (\mathcal{N}^{\text{com}} \mathcal{N}^{\text{com}*})^*$. Further since $\Pi_0 \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}(\mathcal{P})}^* \Pi'_0 \triangleleft \Gamma'$ may not be further split we may assume that for

some $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta'_0)$ we may rewrite the transition as:

$A\beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} c!m\beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{l'}_{\mathcal{G}(\mathcal{P})} \beta''_0 \parallel \Pi_1 \parallel \Pi(c!m \cdot \lambda) \triangleleft \Gamma \oplus \Gamma(c!m \cdot \lambda)$, i.e. $\Pi'_0 \triangleleft \Gamma' = \beta''_0 \parallel \Pi_1 \parallel \Pi(c!m \cdot \lambda) \triangleleft \Gamma \oplus \Gamma(c!m \cdot \lambda)$. Now we know that there is a rule $A \xrightarrow{c!m}_{\mathcal{P}} \epsilon$ in \mathcal{P} and thus $A\beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}} \beta \parallel \Pi_2 \parallel \Pi(c!m) \triangleleft \Gamma \oplus \Gamma(c!m)$ and since $\beta \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta_0$ we may use analogous reasoning to the $\beta' = \epsilon$ case to obtain that $\beta \parallel \Pi_2 \parallel \Pi(c!m) \triangleleft \Gamma \oplus \Gamma(c!m) \xrightarrow{l'}_{\mathcal{P}} \beta'' \parallel \Pi_2 \parallel \Pi(c!m \cdot \lambda) \triangleleft \Gamma \oplus \Gamma(c!m \cdot \lambda)$ such that $\beta'' \parallel \Pi_2 \parallel \Pi(c!m \cdot \lambda) \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta''_0 \parallel \Pi_1 \parallel \Pi(c!m \cdot \lambda)$ which is what we wanted to prove.

- Case: $\beta' = \nu A'$.

Firstly, we notice $\Pi'_1 = \Pi_1$ and $\Gamma = \Gamma_1$ and we may write $\beta_0 = \beta'_0\beta''_0$ such that $\beta'_0 \in \mathcal{N}^{\text{com}*}$ and $\beta''_0 \in (\mathcal{N}^{\neg\text{com}}\mathcal{N}^{\text{com}*})^*$. Further since $\Pi_0 \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}(\mathcal{P})} \Pi'_0 \triangleleft \Gamma'$ may not be further split we may assume that for some $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta'_0)$ we may rewrite the transition as: $A\beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \nu A'\beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{l'}_{\mathcal{G}(\mathcal{P})} \beta''_0 \parallel \Pi_1 \parallel \Pi(\nu A' \cdot \lambda) \triangleleft \Gamma \oplus \Gamma(\nu A' \cdot \lambda)$, i.e. $\Pi'_0 \triangleleft \Gamma' = \beta''_0 \parallel \Pi_1 \parallel \Pi(\nu A' \cdot \lambda) \triangleleft \Gamma \oplus \Gamma(\nu A' \cdot \lambda)$. Now we know that there is a rule $A \xrightarrow{\nu A'}_{\mathcal{P}} \epsilon$ in \mathcal{P} and thus $A\beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}} \beta \parallel \Pi_2 \parallel \Pi(\nu A') \triangleleft \Gamma \oplus \Gamma(\nu A')$ and since $\beta \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta_0$ we may use analogous reasoning to the $\beta' = \epsilon$ case to obtain that $\beta \parallel \Pi_2 \parallel \Pi(\nu A') \triangleleft \Gamma \oplus \Gamma(\nu A') \xrightarrow{l'}_{\mathcal{P}} \beta'' \parallel \Pi_2 \parallel \Pi(\nu A' \cdot \lambda) \triangleleft \Gamma \oplus \Gamma(\nu A' \cdot \lambda)$ such that $\beta'' \parallel \Pi_2 \parallel \Pi(c!m \cdot \lambda) \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta''_0 \parallel \Pi_1 \parallel \Pi(c!m \cdot \lambda)$ which is what we wanted to prove.

- Case: $\beta' = c?m$.

Firstly, we notice $\Pi'_1 = \Pi_1$ and $\Gamma = \Gamma_1$ and we may write $\beta_0 = \beta'_0\beta''_0$ such that $\beta'_0 \in \mathcal{N}^{\text{com}*}$ and $\beta''_0 \in (\mathcal{N}^{\neg\text{com}}\mathcal{N}^{\text{com}*})^*$. Further since $\Pi_0 \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}(\mathcal{P})} \Pi'_0 \triangleleft \Gamma'$ may not be further split we may assume that for some $\lambda \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta'_0)$ we may rewrite the transition as: $A\beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} c?m\beta_0 \parallel \Pi_1 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \beta_0 \parallel \Pi_1 \triangleleft \Gamma \ominus [c \mapsto m] \xrightarrow{l'}_{\mathcal{G}(\mathcal{P})} \beta''_0 \parallel \Pi_1 \parallel \Pi(\lambda) \triangleleft \Gamma \oplus \Gamma(\lambda)$, i.e. $\Pi'_0 \triangleleft \Gamma' = \beta''_0 \parallel \Pi_1 \parallel \Pi(\lambda) \triangleleft \Gamma \ominus [c \mapsto m] \oplus \Gamma(\lambda)$. Now we know that there is a rule $A \xrightarrow{c?m}_{\mathcal{P}} \epsilon$ in \mathcal{P} and thus $A\beta \parallel \Pi_2 \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{P}} \beta \parallel \Pi_2 \triangleleft \Gamma \ominus [c \mapsto m]$ and since $\beta \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta_0$ we may use analogous reasoning to the $\beta' = \epsilon$ case to obtain that $\beta \parallel \Pi_2 \triangleleft \Gamma \ominus [c \mapsto m] \xrightarrow{l'}_{\mathcal{P}} \beta'' \parallel \Pi_2 \parallel \Pi(\lambda) \triangleleft \Gamma \ominus [c \mapsto m] \oplus \Gamma(\lambda)$ such that $\beta'' \parallel \Pi_2 \parallel \Pi(\lambda) \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta''_0 \parallel \Pi_1 \parallel \Pi(\lambda)$ which is what we wanted to prove.

This concludes the proof. \square

Proposition 2. Suppose \mathcal{P} is an ACPS in normal form and $Q = (\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ a simple coverability query. Then, Q is a yes-instance, if and only if, $Q' = (\mathcal{G}(\mathcal{P}), \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ is a yes-instance. Hence simple coverability, boundedness and termination for ACPS and APCPS polynomial-time inter-reduce. A simple APCPS query $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ satisfies: $\pi \simeq_{I(\mathcal{G})} A \in \mathcal{N}$ for all π in Π

and Π_0 . Further \mathcal{P} is K -shaped from $\Pi_0 \triangleleft \Gamma_0$ if, and only if, $\mathcal{G}(\mathcal{P})$ is K -shaped from $\Pi_0 \triangleleft \Gamma_0$.

Proof. First suppose $Q = (\mathcal{P}, \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ is a simple coverability query for ACPS in normal form, $\Pi = A_1 \parallel \dots \parallel A_n$. W.l.o.g. we may assume that all A_i are non-commutative wrt to $I(\mathcal{G}(\mathcal{P}))$ (otherwise we may introduce a fresh channel c_{cov} , message m_{cov} , non-terminals A'_i and local states $0, \dots, K, K+1, \infty$, counting the number of non-commutative non-terminals on stack — widening at $\geq K+2$, rules $(k, A_i) \xrightarrow{\epsilon} A'_i$ for $k \leq K+1$, $A'_i \xrightarrow{c_{\text{cov}}?m_{\text{cov}}} \epsilon$ and change Π to $\Pi = A'_1 \parallel \dots \parallel A'_n$; applying polynomial-time normal form reduction afterwards; this transformation may increase the shaped constraint from K -shaped to $K+1$ -shaped, since (∞, A_i) will clearly be commutative, and is also polynomial).

Defining then $Q' = (\mathcal{G}(\mathcal{P}), \Pi_0 \triangleleft \Gamma_0, \Pi \triangleleft \Gamma)$ is then clearly a simple coverability query and can be polynomial-time computed.

Since $\mathcal{G}(-)$ is clearly a bijection and both $\mathcal{G}(-)$ and $\mathcal{G}(-)^{-1}$ are polynomial time computable we may simply show that Q is a yes-instance if, and only if, Q' is a yes-instance.

Suppose Q' is a yes-instance then $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$ such that $\Pi \triangleleft \Gamma \leq_{\text{APCPS}} \Pi' \triangleleft \Gamma'$. Let us maximally split this path so that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\Pi_1 \triangleleft \Gamma_1}_{\mathcal{G}(\mathcal{P})} \Pi_1 \triangleleft \Gamma_1 \xrightarrow{\Pi_2 \triangleleft \Gamma_2}_{\mathcal{G}(\mathcal{P})} \dots \xrightarrow{\Pi_n \triangleleft \Gamma_n}_{\mathcal{G}(\mathcal{P})} \Pi_n \triangleleft \Gamma_n \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$ using the third labelling of this section. Since Q' is simple we know that $\Pi = A_1 \parallel \dots \parallel A_n$ and thus $\Pi' \simeq_{I(\mathcal{G}(\mathcal{P}))} A_1\beta_1 \parallel \dots \parallel A_n\beta_n \parallel \Pi''$. Further since the path above is a maximal split we may assume that $\Pi_n \simeq_{I(\mathcal{G}(\mathcal{P}))} A_1\beta_1 \parallel \dots \parallel A_n\beta_n \parallel \Pi'''$ and $\Pi''' \triangleleft \Gamma_n \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \Pi'' \triangleleft \Gamma'$ where the latter may not be split into two paths exposing a label. Hence we may conclude that $\Pi''' = B_1\beta_1^\dagger\beta_1^{\dagger'} \parallel \dots \parallel B_k\beta_k^\dagger\beta_k^{\dagger'} \parallel \Pi''''$ where $B_1, \dots, B_k \in \mathcal{N}^{\neg\text{com}}$, $\beta_1^\dagger, \dots, \beta_k^\dagger \in \mathcal{N}^{\text{com}*}$ and $\beta_1^{\dagger'}, \dots, \beta_k^{\dagger'} \in (\mathcal{N}^{\neg\text{com}}\mathcal{N}^{\text{com}*})^*$. that after a one rule step for each process 1 up to k we can go to $\Pi''' \triangleleft \Gamma_n \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \beta_1^{\dagger'}\beta_1^{\dagger'} \parallel \dots \parallel \beta_k^{\dagger'}\beta_k^{\dagger'} \parallel \Pi'''' \triangleleft \Gamma^\dagger \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \Pi'' \triangleleft \Gamma'$ where $\beta_1^{\dagger'}, \dots, \beta_k^{\dagger'} \in \mathcal{N}^{\text{com}*}$, $\Pi'' = \Pi'''' \parallel \Pi(\lambda_1^\dagger \dots \lambda_k^\dagger)$, $\Gamma' = \Gamma^\dagger \oplus \Gamma(\lambda_1^\dagger \dots \lambda_k^\dagger)$ for some $\lambda_i^\dagger \lambda_i^{\dagger'} \in \mathcal{L}_{\mathcal{G}(\mathcal{P})}(\beta_i^{\dagger'})$ for each i .

We will now show that we can follow these transitions with \mathcal{P} . First Lemma 11 tells us that we find the following path $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\Pi_1 \triangleleft \Gamma_1}_{\mathcal{P}} \bar{\Pi}_1 \triangleleft \Gamma_1 \xrightarrow{\Pi_2 \triangleleft \Gamma_2}_{\mathcal{P}} \dots \xrightarrow{\Pi_n \triangleleft \Gamma_n}_{\mathcal{P}} \bar{\Pi}_n \triangleleft \Gamma_n$ such that $\Pi_i \simeq_{I(\mathcal{G}(\mathcal{P}))} \bar{\Pi}_i$ for all $i \in \langle n \rangle$. Now $\bar{\Pi}_n = A_1\bar{\beta}_1 \parallel \dots \parallel A_n\bar{\beta}_n \parallel \bar{\Pi}'''$ and $\Pi'''' = B_1\bar{\beta}_1^\dagger\bar{\beta}_1^{\dagger'} \parallel \dots \parallel B_k\bar{\beta}_k^\dagger\bar{\beta}_k^{\dagger'} \parallel \bar{\Pi}''''$ since $A_1, \dots, A_n, B_1, \dots, B_k \in \mathcal{N}^{\neg\text{com}}$. Now since $B_1, \dots, B_k \in \mathcal{N}^{\neg\text{com}}$ it is easy to see that we can follow the $\mathcal{G}(\mathcal{P})$ -path one rule step per process in: $\bar{\Pi}''' \triangleleft \Gamma_n \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \bar{\beta}_1^{\dagger'}\bar{\beta}_1^{\dagger'} \parallel \dots \parallel \bar{\beta}_k^{\dagger'}\bar{\beta}_k^{\dagger'} \parallel \bar{\Pi}'''' \triangleleft \Gamma^\dagger$ where $\bar{\beta}_1^{\dagger'}, \dots, \bar{\beta}_k^{\dagger'} \in \mathcal{N}^{\text{com}*}$ and $\bar{\beta}_i^{\dagger'} \simeq_{I(\mathcal{G}(\mathcal{P}))} \beta_i^{\dagger'}$ for all $i \in \langle k \rangle$. We can then pick $\bar{\lambda}_i^\dagger \in \mathcal{L}_{\mathcal{P}}(\bar{\beta}_i^{\dagger'})$ such that $\bar{\lambda}_i^\dagger \simeq_{I(\mathcal{G}(\mathcal{P}))} \lambda_i^\dagger\lambda_i^{\dagger'}$. Lemma 10 then tells us we can reach $\bar{\Pi}''' \triangleleft \Gamma_n \xrightarrow{\epsilon}_{\mathcal{G}(\mathcal{P})} \bar{\Pi}^\dagger \parallel \bar{\Pi}'''' \parallel \Pi(\bar{\lambda}_1^\dagger \dots \bar{\lambda}_k^\dagger) \triangleleft \Gamma^\dagger \oplus \Gamma^\dagger \oplus \Gamma(\bar{\lambda}_1^\dagger \dots \bar{\lambda}_k^\dagger)$ where we write $\bar{\Pi}^\dagger = \bar{\beta}_1^{\dagger'} \parallel \dots \parallel \bar{\beta}_k^{\dagger'}$. We

note that $\Gamma \leq \Gamma(\lambda_1^\dagger \cdots \lambda_k^\dagger) \leq \Gamma(\bar{\lambda}_1^\dagger \cdots \bar{\lambda}_k^\dagger)$. We can now use these paths to extend our \mathcal{P} path to a covering configuration: $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{P}} \bar{\Pi}_n \triangleleft \Gamma_n \xrightarrow{\mathcal{P}} A_1 \beta_1 \parallel \cdots \parallel A_n \beta_n \parallel \bar{\Pi}^\dagger \parallel \bar{\Pi}''' \parallel \Pi(\bar{\lambda}_1^\dagger \cdots \bar{\lambda}_k^\dagger) \triangleleft \Gamma^\dagger \oplus \Gamma^\dagger \oplus \Gamma(\bar{\lambda}_1^\dagger \cdots \bar{\lambda}_k^\dagger)$ which clearly implies that $\Pi \triangleleft \Gamma$ is coverable in \mathcal{P} from $\Pi_0 \triangleleft \Gamma_0$. Hence Q is a yes-instance.

Conversely, suppose Q is a yes-instance. Then $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{P}} \Pi' \triangleleft \Gamma'$ such that $\Pi \triangleleft \Gamma \leq_{ACPS} \Pi' \triangleleft \Gamma'$. Since Q is a simple query we know that $\Pi = A_1 \parallel \cdots \parallel A_n$ and hence we may conclude that $\Pi' = A_1 \beta_1 \parallel \cdots \parallel A_n \beta_n \parallel \Pi''$. Lemma 9 then allows us to conclude that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{G}(\mathcal{P})} \Pi'_0 \triangleleft \Gamma'$ where $\Pi'_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi'$ i.e. $\Pi'_0 \simeq_{I(\mathcal{G}(\mathcal{P}))} A_1 \beta_1 \parallel \cdots \parallel A_n \beta_n \parallel \Pi''$ and thus clearly $\Pi \triangleleft \Gamma \leq_{APCPS} \Pi'_0 \triangleleft \Gamma'$ which implies Q' is a yes-instance.

For the second claim, we will prove that there is a reachable K -shaped configuration from $\Pi_0 \triangleleft \Gamma_0$ in \mathcal{P} if, and only if, there is a reachable K -shaped configuration from $\Pi_0 \triangleleft \Gamma_0$ in $\mathcal{G}(\mathcal{P})$.

Suppose then that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{P}} \Pi \triangleleft \Gamma$ and that $\Pi \triangleleft \Gamma$ is K -shaped. An application of Lemma 9 then yields that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{G}(\mathcal{P})} \Pi' \triangleleft \Gamma'$ where $\Pi' \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi$ which clearly implies that Π' is K -shaped as processes of Π' are only (commutative) permutations of processes in Π . Conversely, suppose $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{G}(\mathcal{P})} \Pi \triangleleft \Gamma$ and $\Pi \triangleleft \Gamma$ is K -shaped. As before let us maximally split this path so that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\Pi_1 \triangleleft \Gamma_1}^* \mathcal{G}(\mathcal{P})$ $\Pi_1 \triangleleft \Gamma_1 \xrightarrow{\Pi_2 \triangleleft \Gamma_2}^* \mathcal{G}(\mathcal{P}) \cdots \xrightarrow{\Pi_n \triangleleft \Gamma_n}^* \mathcal{G}(\mathcal{P})$ $\Pi_n \triangleleft \Gamma_n \xrightarrow{\epsilon}^* \mathcal{G}(\mathcal{P})$ $\Pi \triangleleft \Gamma$ using the third labelling of this section. Now $\Pi = \beta_1 \beta'_1 \parallel \cdots \parallel \beta_n \beta'_n$ with $\beta_i \in \mathcal{N}^{\text{com}*}$ and $\beta'_i \in (\mathcal{N}^{\text{com}*} \mathcal{N}^{\text{com}*})^*$. Using Lemma 10 we know that we can extend the path above to $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{G}(\mathcal{P})} \Pi \triangleleft \Gamma \xrightarrow{\mathcal{G}(\mathcal{P})} \beta'_1 \parallel \cdots \parallel \beta'_n \triangleleft \Gamma' =: \Pi' \triangleleft \Gamma'$. Clearly $\beta'_1 \parallel \cdots \parallel \beta'_n \in \mathbb{M}[\mathcal{N}^{\text{com}*} \mathcal{N}^*]$ and since $\Pi \triangleleft \Gamma$, and we have only “executed off” all commutative β_i ’s, it is easy to see that $\Pi' \triangleleft \Gamma'$ is also K -shaped. Since $\beta'_1 \parallel \cdots \parallel \beta'_n \in \mathbb{M}[\mathcal{N}^{\text{com}*} \mathcal{N}^*]$ we can invoke Lemma 11 to see $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\mathcal{P}} \bar{\Pi}' \triangleleft \Gamma'$ such that $\bar{\Pi}' \simeq_{I(\mathcal{G}(\mathcal{P}))} \Pi'$. Hence as above we may conclude that $\bar{\Pi}' \triangleleft \Gamma'$ is K -shaped.

Hence we can conclude that there is a reachable K -shaped configuration from $\Pi_0 \triangleleft \Gamma_0$ in \mathcal{P} if, and only if, there is a reachable K -shaped configuration from $\Pi_0 \triangleleft \Gamma_0$ in $\mathcal{G}(\mathcal{P})$. From which we can deduce that \mathcal{P} is K -shaped if, and only if, $\mathcal{G}(\mathcal{P})$ is K -shaped. \square

B. Proofs for Section III

Lemma 2. $(\mathcal{M}^{\text{col}}, \leq_{\mathcal{M}^{\text{col}}})$ and $(\text{Config}, \leq_{\text{Config}})$ are WQO.

Proof. Let for all $P \subseteq \mathcal{P}^{\text{col}}$ define $M(P)$ to be set of complex tokens that exactly contain coloured tokens only of colours not in P , i.e. $M(P) = \{m \in \mathcal{M}^{\text{col}} : \forall p \in P. m(p) = \emptyset, \forall p' \notin P. m(p') \neq \emptyset\}$. It should be clear that for each $m \in \mathcal{M}^{\text{col}}$ there is a unique $P \subseteq \mathcal{P}^{\text{col}}$ such that $m \in M(P)$. Hence we can see that \mathcal{M}^{col} is isomorphic to the finite disjoint union $\sum_{P \subseteq \mathcal{P}^{\text{col}}} M(P)$. Let $P \subseteq \mathcal{P}^{\text{col}}$ we can see that $M(P)$ is isomorphic to $\prod_{p \in P} \{\emptyset\} \times \prod_{p \notin P} \mathbb{M}\{\bullet\}$ and so clearly

$\leq_{M(P)} := \prod_{p \in P} = \times \prod_{p \notin P} \subseteq_{\mathbb{M}\{\bullet\}}$ is a WQO (eliding the isomorphism) for $M(P)$. Hence $\sum_{P \subseteq \mathcal{P}^{\text{col}}} \leq_{M(P)}$ is a WQO for $\sum_{P \subseteq \mathcal{P}^{\text{col}}} M(P)$ and thus for \mathcal{M}^{col} (eliding the isomorphism).

Suppose $m, m' \in \mathcal{M}^{\text{col}}$ such that $m \leq_{\mathcal{M}^{\text{col}}} m'$. Let $P = \{p : m(p) = \emptyset\}$ since $m \leq_{\mathcal{M}^{\text{col}}} m'$ it is easy to see that $P = \{p : m'(p) = \emptyset\}$. Hence $m, m' \in M(P)$ and we also know that for all $p \notin P$ we have $0 < |m(p)| \leq |m'(p)|$, i.e. $m(p) \subseteq_{\mathbb{M}\{\bullet\}} m'(p)$, hence $m \leq_{M(P)} m'$ and thus $m \leq_{\sum_{P \subseteq \mathcal{P}^{\text{col}}} M(P)} m'$. In the opposite direction suppose m, m' are such $m \leq_{\sum_{P \subseteq \mathcal{P}^{\text{col}}} M(P)} m'$ then both $m, m' \in M(P)$ for some $P \subseteq \mathcal{P}^{\text{col}}$ and $m \leq_{M(P)} m'$ which means that for all $p \in P$ it is the case that $0 = |m(p)| = |m'(p)| = |\emptyset|$ and for all $p \notin P$ we have both $m(p), m'(p) \neq \emptyset$ and $m(p) \subseteq_{\mathbb{M}\{\bullet\}} m'(p)$, i.e. $0 < |m(p)| \leq |m'(p)|$ and hence $m \leq_{\mathcal{M}^{\text{col}}} m'$. $(\mathcal{M}^{\text{col}}, \leq_{\mathcal{M}^{\text{col}}})$ is then isomorphic to a WQO and hence is a WQO.

For $(\text{Config}, \leq_{\text{Config}})$. It is clear that $(\mathbb{M}\{\bullet\}, \subseteq_{\mathbb{M}\{\bullet\}})$ is a WQO; since \mathcal{P}^{col} is finite, Dickson’s lemma tells us that $(\mathcal{M}^{\text{simple}}, \leq_{\mathcal{M}^{\text{simple}}})$ is a WQO where $\leq_{\mathcal{M}^{\text{simple}}}$ is the extension from $\subseteq_{\mathbb{M}\{\bullet\}}$ to $\subseteq_{\mathcal{P}^{\text{col}} \rightarrow \mathbb{M}\{\bullet\}}$. Further since multi sets over a WQO are again a WQO, $(\mathbb{M}[\mathcal{M}^{\text{col}}], \subseteq_{\mathbb{M}[\mathcal{M}^{\text{col}}]})$ is a WQO and so, we can infer that $(\text{Config}, \leq_{\text{Config}})$ is a WQO which concludes the proof. \square

Lemma 12. \oplus is monotone in both arguments with respect to \leq_{Config} and $\leq_{\mathcal{M}^{\text{col}}}$.

Proof. Suppose we have $m, m', m_0, m'_0 \in \mathcal{M}^{\text{col}}$ such that $m \leq_{\mathcal{M}^{\text{col}}} m_0$ and $m' \leq_{\mathcal{M}^{\text{col}}} m'_0$. Let $p \in \mathcal{P}^{\text{col}}$. If $|(m \oplus m')(p)| = 0$ we can conclude that $|m(p)| = |m'(p)| = 0$ then $\leq_{\mathcal{M}^{\text{col}}}$ guarantees that $|m(p)| = |m_0(p)| = 0$ and $|m'(p)| = |m'_0(p)| = 0$ and hence $|(m_0 \oplus m'_0)(p)| = 0$.

Otherwise either $|m(p)|$ or $|m'(p)| > 0$. W.l.o.g. assume $|m(p)| > 0$ then we know that $|m(p)| \leq_{\mathcal{M}^{\text{col}}} |m_0(p)|$ and $|m_0(p)| > 0$. Further we can easily see that $|m'(p)| \leq |m'_0(p)|$. Hence we can see that $0 < |(m \oplus m')(p)| \leq |(m_0 \oplus m'_0)(p)|$. Hence we can deduce that $m \oplus m' \leq_{\mathcal{M}^{\text{col}}} m_0 \oplus m'_0$ which is what we want to prove.

Suppose we have $s, s', s_0, s'_0 \in \text{Config}$ and $s \leq_{\text{Config}} s_0$ and $s' \leq_{\text{Config}} s'_0$. Let $p \in \mathcal{P}^{\text{col}}$ then clearly

$$|(s \oplus s')(p)| = |s(p)| + |s'(p)| \leq |s_0(p)| + |s'_0(p)| = |(s_0 \oplus s'_0)(p)|.$$

Further let $p' \in \mathcal{P}^{\text{c}}$ then

$$(s \oplus s')(p') = [m_1, \dots, m_n, m'_1, \dots, m'_{n'}] \text{ and} \\ (s_0 \oplus s'_0)(p') = [M_1, \dots, M_N, M'_1, \dots, M'_{N'}]$$

where we may assume:

$$s(p') = [m_1, \dots, m_n], \quad s'(p') = [m'_1, \dots, m'_{n'}], \\ s_0(p') = [M_1, \dots, M_N] \text{ and } s'_0(p') = [M'_1, \dots, M'_{N'}].$$

Further since $s \leq_{\text{Config}} s_0$ and $s' \leq_{\text{Config}} s'_0$ we know that $n \leq N$ and $n' \leq N'$ and for all $i \in \langle n \rangle$ we have $m_i \leq_{\mathcal{M}^{\text{col}}} M_i$ and $m'_j \leq_{\mathcal{M}^{\text{col}}} M'_j$ for all $j \in \langle n' \rangle$. This gives us an injection that pairs up m_i with M_i and m'_j with M'_j for $i \in \langle n \rangle$ and $j \in \langle n' \rangle$.

We can thus conclude that $(s \oplus s')(p') \leq_{\mathbb{M}[\mathcal{M}^{col}]} (s_0 \oplus s'_0)(p')$ which of course implies $s \oplus s' \leq_{Config} s_0 \oplus s'_0$. \square

Lemma 13. Suppose we have $s, s_0, s' \in Config$, $s \leq_{Config} s_0$ and $s'(p)(m) = 0$ if both $p \in \mathcal{P}^C$ and $m \neq \mathbf{0}$ then $s \ominus s' \leq s_0 \ominus s'$.

Proof. Let $p \in \mathcal{P}^S$ then clearly

$$|(s \ominus s')(p)| = |s(p)| - |s'(p)| \leq |s_0(p)| - |s'(p)| = |(s_0 \ominus s')(p)|.$$

Further let $p' \in \mathcal{P}^C$ then

$$\begin{aligned} (s \ominus s')(p')(\mathbf{0}) &= s(p')(\mathbf{0}) - s'(p')(\mathbf{0}) \\ &\leq s_0(p')(\mathbf{0}) - s'(p')(\mathbf{0}) \\ &= (s_0 \ominus s')(p')(\mathbf{0}) \\ (s \ominus s')(p') \upharpoonright (\mathbb{M}[\mathcal{M}^{col}] \setminus \{\mathbf{0}\}) \\ &= s(p') \upharpoonright (\mathbb{M}[\mathcal{M}^{col}] \setminus \{\mathbf{0}\}) \\ &\leq s_0(p') \upharpoonright (\mathbb{M}[\mathcal{M}^{col}] \setminus \{\mathbf{0}\}) \\ &= (s_0 \ominus s')(p') \upharpoonright (\mathbb{M}[\mathcal{M}^{col}] \setminus \{\mathbf{0}\}) \end{aligned}$$

Hence $s \oplus s' \leq_{Config} s_0 \oplus s'_0$. \square

Lemma 14. Let $p \in \mathcal{P}^C$ and $s, s' \in Config$. Suppose $m, m' \in \mathcal{M}^{col}$ such that $m' = \min \{m_0 \in s'(p) : m \leq_{\mathcal{M}^{col}} m_0\}$. If $s \oplus [p \mapsto m] \leq_{Config} s' \oplus [p \mapsto m']$ then $s \leq_{Config} s'$.

Proof. Let $p \in \mathcal{P}^S$ then clearly

$$|s(p)| = |(s \oplus [p \mapsto m])(p)| \leq |(s' \oplus [p \mapsto m'])(p)| = |s'(p)|.$$

Further let $p' \in \mathcal{P}^C$ such that $p \neq p'$ then

$$\begin{aligned} s(p') &= (s \oplus [p \mapsto m])(p') \leq_{\mathbb{M}[\mathcal{M}^{col}]} (s' \oplus [p \mapsto m])(p') \\ &= s'(p'). \end{aligned}$$

Focussing on p we see:

$$s(p) = [m_1, \dots, m_n], \quad s'(p) = [m'_1, \dots, m'_{n'}],$$

where we know that

$$\begin{aligned} (s \oplus [p \mapsto m])(p) &= [m_1, \dots, m_n, m] \text{ and} \\ (s_0 \oplus [p \mapsto m'])(p) &= [m'_1, \dots, m'_{n'}, m'] \end{aligned}$$

where $n \leq n'$ and there is an bijection h from $M := [m_1, \dots, m_n, m]$ to $M' := [m'_1, \dots, m'_{n'}, m']$. such that $m^0 \leq_{\mathcal{M}^{col}} h(m)^0$ for all $m^0 \in M$. Suppose h pairs up m with m' then clearly h is an injection justifying $s(p) \leq_{\mathbb{M}[\mathcal{M}^{col}]} s'(p)$. Otherwise say $h(m_i) = m'$ then since $m \leq_{\mathcal{M}^{col}} h(m)$ we know that $m' \leq_{\mathcal{M}^{col}} h(m)$ since m' is a minimum. Thus $h[m_i \mapsto h(m)]$ is an injection justifying $s(p) \leq_{\mathbb{M}[\mathcal{M}^{col}]} s'(p)$. We can thus conclude $s \leq_{Config} s'$. \square

Lemma 15. \upharpoonright is monotone in the first argument with respect to $\leq_{\mathcal{M}^{col}}$.

Proof. Let $m, m' \in \mathcal{M}^{col}$ such that $m \leq_{\mathcal{M}^{col}} m'$ and $P \subseteq \mathcal{P}^{col}$. Suppose $p \in P$ and $0 < m(p)$ then $0 < |m(p)| = |(m \upharpoonright P)(p)| = |m(p)| \leq |m'(p)| = |(m' \upharpoonright P)(p)|$. If $p \in P$ and $|m(p)| = 0$ then $0 = |m(p)| = |(m \upharpoonright P)(p)| =$

$|m(p)| = |m'(p)| = |(m' \upharpoonright P)(p)|$. Otherwise $p' \notin P$ then $|m \upharpoonright P|(p') = 0 = |(m' \upharpoonright P)(p')|$. Hence clearly $m \upharpoonright P \leq_{\mathcal{M}^{col}} m' \upharpoonright P$. \square

Proposition 3. Given an NNCT \mathcal{N} , $(Config, \rightarrow_{\mathcal{N}}, \leq_{Config})$ is a strict WSTS.

Proof. Let $\mathcal{N} = (\mathcal{P}^S, \mathcal{P}^C, \mathcal{P}^{col}, \mathcal{R}, \zeta)$ and suppose $s, s' \in Config$ such that $s <_{Config} s'$. Further suppose we can make the transition $s \xrightarrow{r}_{\mathcal{N}} t$ using rule $r \in \mathcal{R}$. Let us perform a case analysis on r

- *Case:* $r = (I, O) \in SimpleRules$.

Since r is enabled at s we know that $s \ominus I \in Config$. Lemma 13 then yields that $s \ominus I \leq_{Config} s' \ominus I$ and hence clearly r is enabled at s' . Thus $s \xrightarrow{r}_{\mathcal{N}} s' \ominus I \oplus O =: t'$. Since $t = s \ominus I \oplus O$ Lemma 12 gives us $t \leq_{Config} t'$. Since $s \neq s'$ it is clear that $t \neq t'$ and hence we obtain $t <_{Config} t'$ which is what we want to prove.

- *Case:* $r = ((p, I), (p', c, O)) \in ComplexRules$.

Since r is enabled at s we know that for some $m \in s(p)$ $s \ominus I [p \mapsto m] \in Config$ and

$$t = s \ominus I [p \mapsto m] \oplus O \oplus [p' \mapsto m \oplus c].$$

First Lemma 13 then yields that $s \ominus I \leq_{Config} s' \ominus I$. Since $s <_{Config} s'$ there exists $m' \in s'(p)$ such that $m \leq_{\mathcal{M}^{col}} m'$; w.l.o.g. we can assume that $m' = \min \{m_0 \in s'(p) : m \leq_{\mathcal{M}^{col}} m_0\}$. Since $I \in Config_S$ it is also the case that $m' = \min \{m_0 \in s'(p) \ominus I : m \leq_{\mathcal{M}^{col}} m_0\}$. Lemma 14 then yields that $s \ominus I [p \mapsto m] \leq_{Config} s' \ominus I [p \mapsto m']$ hence it is easy to see that r is enabled at s' . Further

$$s' \xrightarrow{r}_{\mathcal{N}} s' \ominus I [p \mapsto m'] \oplus O \oplus [p' \mapsto m' \oplus c] =: t'.$$

Since $m \leq_{\mathcal{M}^{col}} m'$ Lemma 12 yields $m \oplus c \leq_{\mathcal{M}^{col}} m' \oplus c$. Hence it is easy to see $[p' \mapsto m \oplus c] \leq_{Config} [p' \mapsto m' \oplus c]$. Lemma 12 then gives us that $t \leq_{Config} t'$. Since $s \neq s'$ either $s' \ominus I [p \mapsto m'] \neq s \ominus I [p \mapsto m]$ or $m \neq m'$. Noting this we can see that $t \neq t'$ and hence $t <_{Config} t'$ which is what we want to prove.

- *Case:* $r = ((p, I), (p', P, O)) \in TransferRules$.

Since r is enabled at s we know that for some $m \in s(p)$ $s \ominus I [p \mapsto m] \in Config$ and

$$t = s \ominus I [p \mapsto m] \oplus O \oplus [p' \mapsto m_{\overline{P}}] \oplus (m_P \circ \zeta^{-1})$$

where $m_P = m \upharpoonright P$ and $m_{\overline{P}} = m \upharpoonright (\mathcal{P}^{col} \setminus P)$. First Lemma 13 then yields that $s \ominus I \leq_{Config} s' \ominus I$. Since $s <_{Config} s'$ there exists $m' \in s'(p)$ such that $m \leq_{\mathcal{M}^{col}} m'$; w.l.o.g. we can assume that $m' = \min \{m_0 \in s'(p) : m \leq_{\mathcal{M}^{col}} m_0\}$. Since $I \in Config_S$ it is also the case that $m' = \min \{m_0 \in s'(p) \ominus I : m \leq_{\mathcal{M}^{col}} m_0\}$. Lemma 14 then yields that $s \ominus I [p \mapsto m] \leq_{Config} s' \ominus I [p \mapsto m']$ hence it is easy to see that r is enabled at s' . Further

$$s' \xrightarrow{r}_{\mathcal{N}} s' \ominus I [p \mapsto m'] \oplus O \oplus [p' \mapsto m'_{\overline{P}}] \oplus (m'_P \circ \zeta^{-1}) =: t'.$$

where $m_P = m \upharpoonright P$ and $m_{\overline{P}} = m \upharpoonright (\mathcal{P}^{col} \setminus P)$. Since $m \leq_{\mathcal{M}^{col}} m'$ Lemma 15 yields both $m_P \leq_{\mathcal{M}^{col}} m'_P$ and $m_{\overline{P}} \leq_{\mathcal{M}^{col}} m'_{\overline{P}}$. Hence it is easy to see $[p' \mapsto m_{\overline{P}}] \leq_{Config}$

$[p \mapsto m'_P]$. Further clearly $|(m_P \circ \zeta^{-1})(p)| \leq |(m_P \circ \zeta^{-1})(p)|$ for all $p \in \mathcal{P}^S$ and thus $(m_P \circ \zeta^{-1}) \leq_{Config} (m_P \circ \zeta^{-1})$. Lemma 12 then gives us that $t \leq_{Config} t'$. Since $s \neq s'$ either $s' \ominus I[p \mapsto m'] \neq s \ominus I[p \mapsto m]$ or $m \neq m'$. The later implies that either $m_P \neq m'_P$ or $m_{\bar{P}} \neq m'_{\bar{P}}$. Noting this we can see that $t \neq t'$ and hence $t <_{Config} t'$ which is what we want to prove. \square

Theorem 3. *Simple coverability for K-shaped APCPS in the alternative semantics EXPTIME-time reduces to NNCT coverability.*

Proof. Fix a K-shaped APCPS $\mathcal{G} = (\Sigma, I, \mathcal{N}, \mathcal{R}, S)$ from $\Pi_0 \triangleleft \Gamma_0$ where $K \geq 1$ and a simple coverability query $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, A_1^{\text{cov}} \parallel \dots \parallel A_n^{\text{cov}} \triangleleft \Gamma^{\text{cov}})$. We first define a simulating NNCT $\mathcal{N} = (\mathcal{P}^S, \mathcal{P}^C, \mathcal{P}^{\text{col}}, \mathcal{R}, \zeta)$.

- For each $msg \in \text{Msg}$ and $c \in \text{Chan}$, we introduce a simple place $p_{c, msg}^S$.
- For each $X \in \mathcal{N}$, we introduce a simple place $p_{\nu X}^S$.
- For each $0 \leq k \leq K$, $X_k \dots X_1 \in (\mathcal{N}^{\text{-com}})^*$, $\ell_{k+1} \ell_k \dots \ell_1 \in (\{A_i^{\text{cov}} : i \in \langle n \rangle\} \cup \{+, -\})^*$ and $\$ \in \Sigma \cup \{\epsilon\} \cup \mathcal{N}$, we introduce a complex place $p_{\$, \ell_{k+1} X_k \ell_k \dots X_1 \ell_1}^C$; we also introduce a complex place $p_{\$, (+)^N \ell_{k+1} X_k \ell_k \dots X_1 \ell_1}^C$ for each $\$' \in \Sigma \cup \{\epsilon\}$ and $N \in \mathcal{N}$.
- We introduce an auxiliary simple place p_{budget}^{CCFG} and for each $X \in \mathcal{N}$, we introduce a simple place p_X^{CCFG} ; for each $0 \leq k \leq K$, $X_k \dots X_1 \in (\mathcal{N}^{\text{-com}})^*$, $\ell_{k+1} \ell_k \dots \ell_1 \in (\mathcal{P}[\{A_i^{\text{cov}} : i \in \langle n \rangle\} \cup \{+\}])^*$ and $\$ \in \Sigma \cup \{\epsilon\} \cup \mathcal{N}$, we introduce a complex place $p_{\$, \ell_{k+1} X_k \ell_k \dots X_1 \ell_1}^{CCFG}$, which will be used by \mathcal{N} to implement a CCFG widget.
- For each non-terminal A_i^{cov} in the coverability query where $i \in \langle n \rangle$ \mathcal{N} has a simple place $p_{A_i}^{\text{cov}}$.
- The NNCT \mathcal{N} will further have four special simple places: p^{sim} , p^{CCFG} , p^{CCFG+} and p^{Query} .
- For each $1 \leq k \leq K + 1$ and $e \in \Sigma^{\text{com}}$, we introduce a colour $p_{k, e}^I$.
- We define a map $\zeta : \mathcal{P}^{\text{col}} \rightarrow \mathcal{P}^S$ by $p_{k, c!msg}^I \mapsto p_{c, msg}^S$ and $p_{k, \nu X}^I \mapsto p_{\nu X}^S$.

Let us further define three special simple markings $s_{\text{sim}} = [p^{\text{sim}} \mapsto [\bullet]]$, $s_{CCFG} = [p^{CCFG} \mapsto [\bullet]]$, $s_{CCFG+} = [p^{CCFG+} \mapsto [\bullet]]$ and $s_{\text{Query}} = [p^{\text{Query}} \mapsto [\bullet]]$.

An APCPS configuration $\Pi \triangleleft \Gamma \in \text{Config}^{APCPS}$ is represented as an NNCT configuration as follows:

- For each $c \in \text{Chan}$ and $msg \in \text{Msg}$ place $p_{c, msg}^S$ contains precisely one \bullet -token for each occurrence of msg in c — we can formalise this as a function $\mathcal{F}^{\Gamma}(\Gamma) = [p_{c, msg}^S \mapsto [\bullet^{\Gamma(c)(msg)}] \mid c \in \text{Chan}, msg \in \text{Msg}]$.
- Let $0 \leq k \leq K + 1$. The representation of the state of a process $\pi = \$M_{k+1}\gamma \in \Pi$ with $\gamma = X_k M_k \dots X_1 M_1$ is defined by a case analysis in three cases — a general case and two edge cases:

- If either $0 < k \leq K$ or $k = 0$ and it is not the case that both $\$ \in \mathcal{N}$ and $M_1 = \emptyset$ then we represent π as follows: (i) for each $e \in M_i$ there is one $p_{i, e}^I$ -coloured token in m where $1 \leq i \leq k + 1$ and $e \in \Sigma^{\text{com}}$ — or equivalently $m = \mathcal{F}^I(M_1, \dots, M_{k+1})$ where we define the function $\mathcal{F}^I(M_1, \dots, M_{k+1}) = [p_{i, e}^I \mapsto [\bullet^{M_i(e)}] \mid 1 \leq i \leq k + 1, e \in \Sigma^{\text{com}}]$; (ii) for each $1 \leq i \leq k + 1$, if the cache $M_i \in \text{TermCache}$ then let $\ell_i = +$; otherwise let $\ell_i \in \{-\} \cup \{A_j^{\text{cov}} \mid j \in \langle n \rangle, M_i(A_j^{\text{cov}}) > 0\}$. We refer to a possible value of ℓ_i as a *character* of M_i . The sequence of $\$,$ the non-commutative non-terminals $X_k \dots X_1$ and a

possible choice of characters $\ell_{k+1}, \dots, \ell_1$ of M_{k+1}, \dots, M_1 is represented as the complex place in which m located, i.e. m is placed in $p_{\$, \ell_{k+1} X_k \ell_k \dots X_1 \ell_1}^C$. We formalise the representation of one process as the set of markings $\mathcal{F}^\pi(\$ M_{k+1} X_k M_k \dots X_1 M_1) = \{[p_{\$, \ell_{k+1} X_k \ell_k \dots X_1 \ell_1}^C \mapsto [\mathcal{F}^i(M_1, \dots, M_{k+1})]] \mid \ell_i \text{ character of } M_i, i \in \langle k+1 \rangle\}$, where the possible characters of M_{k+1}, \dots, M_1 can be thought of non-deterministically chosen.

◦ If $k = 0$, $\$ \in \mathcal{N}$, and $M_1 = \emptyset$ then we may represent π in addition to the representation above also as a \bullet token in $p_{\nu \S hence the representation of π is defined as $\mathcal{F}^\pi(\pi) = \{[p_{\nu \$}^S \mapsto [\bullet]]\} \cup \mathcal{F}^\pi(\$ \emptyset)$ where $\mathcal{F}^\pi(\$ \emptyset)$ is as defined in the general case above.

◦ If $k = K + 1$ and $\pi = \$ M_{K+2} \gamma \in \Pi$ with $\gamma = X_{K+1} M_{K+1} X_K M_K \dots X_1 M_1$ then since \mathcal{G} is a K -shaped APCPS it will be the case that $M_{K+2} = \emptyset$ and $X_{K+1} \in \mathcal{N}$ and $\$ \in \Sigma \cup \epsilon$. We notice that any character for M_{k+2} must be $+$ and so $\$ M_{K+2} \gamma$ is represented by a complex token $m = \mathcal{F}^i(M_1, \dots, M_{K+2}) := \mathcal{F}^i(M_1, \dots, M_{K+1})$ and with the set of markings $\mathcal{F}^\pi(\$ M_{K+2} \gamma) = \{[p_{\$, (+) X_{K+1} \ell_{K+1} X_K \ell_K \dots X_1 \ell_1}^C \mapsto [\mathcal{F}^i(M_1, \dots, M_{K+2})]] \mid \ell_i \text{ character of } M_i, i \in \langle k+1 \rangle\}$. For uniformity we will not treat this special case explicitly in the following but we will note that this special representation applies when $k = K + 1$.

Representing all processes in $\Pi = \pi_1 \parallel \dots \parallel \pi_n$ can then be formalised as

$$\mathcal{F}^\Pi(\Pi) = \left\{ \bigoplus_{i=1}^n s_i \mid s_i \in \mathcal{F}^\pi(\pi_i) \right\}.$$

The representation of $\Pi \triangleleft \Gamma$ is a set of configurations where Γ is represented in \mathcal{N} 's simple places by $\mathcal{F}^\Gamma(\Gamma)$ and Π is represented in \mathcal{N} 's complex and simple places by a configuration from $\mathcal{F}^\Pi(\Pi)$ together with a \bullet -token in p_{sim} or formally as: $\mathcal{F}(\Pi \triangleleft \Gamma) = \{s \oplus (\mathcal{F}^\Gamma(\Gamma) \oplus s_{sim}) \mid s \in \mathcal{F}^\Pi(\Pi)\}$. We can define the relation $R \subseteq \text{Config}^{APCPS} \times \mathcal{P}[\text{Config}]$ by

$$R = \left\{ (\Pi \triangleleft \Gamma, \mathcal{F}(\Pi \triangleleft \Gamma)) \mid \Pi \triangleleft \Gamma \in \text{Config}^{APCPS} \right\};$$

we will prove in the following that R is a weak bisimulation between a labelled version of $(\text{Config}^{APCPS}, \rightarrow_{\text{con}'})$ and a labelled version of a “co-universal powerset lifting” of $(\text{Config}, \rightarrow_{\mathcal{N}})$.

Let us turn to the implementation of the alternative semantics as defined in the right column of Table I in \mathcal{N} 's rules. In the following we write $\Xi = X_k \ell_k \dots X_1 \ell_1$ to save space.

The alternative semantics defines transitions of the form $\gamma \parallel \Pi \triangleleft \Gamma \rightarrow_{\text{con}'} \gamma' \parallel \Pi' \triangleleft \Gamma'$. We will describe \mathcal{N} 's rules by a case analysis on the rule that justifies the transition and relate the forms of $\gamma, \gamma', \Pi, \Pi', \Gamma$ and Γ' to guide the reader's intuition.

• **Rule (R'-1):** $\Pi = \Pi', \Gamma = \Gamma'$ and $A \rightarrow \beta$ is a \mathcal{G} rule.

We perform case analysis on β :

- $\beta = BC$ and C non-commutative.

For each sequence of non-commutative non-terminals and

cache characters $A \ell_{k+1} X_k \ell_k \dots X_1 \ell_1$ where $k < K$ we introduce a complex rule $((p_{A, \ell_{k+1} \Xi}^C, s_{sim}), (p_{B, (+) C \ell_{k+1} \Xi}^C, \emptyset, s_{sim}))$ that moves a complex token from a place encoding $A \ell_{k+1} \Xi$ to a place representing $B (+) C \ell_{k+1} \Xi$.

- $\beta = a$ where $a \in \Sigma \cup \{\epsilon\}$

For each sequence of non-commutative non-terminals and cache characters $A \ell_{k+1} X_k \ell_k \dots X_1 \ell_1$ where $k \leq K$ we introduce a complex rule $((p_{A, \ell_{k+1} \Xi}^C, s_{sim}), (p_{a, \ell_{k+1} \Xi}^C, \emptyset, s_{sim}))$ that moves a complex token from a place encoding $A \ell_{k+1} \Xi$ to a place representing $a \ell_{k+1} \Xi$.

• **Rule (R'-2):** $A \rightarrow BC$ is a \mathcal{G} -rule and C commutative.

Note that a reduction $C \rightarrow_{\text{seq}}^* w \in (\mathcal{N}^{\text{com}} \cup \Sigma^{\text{com}})^*$ is that of a commutative context-free grammar (CCFG) for which a Petri-net encoding exists [22] by Ganty and Majumdar which builds on an earlier encoding [16] by Esparza. Ganty and Majumdar leverage a recent result [19]: every word of a CCFG has a *bounded-index derivation*. The CCFG widget can thus be augmented with a budget counter that ensures that the Petri-net encoding respects boundedness of index. Termination of such a CCFG computation is signaled by a transition which is only enabled when the budget counter reaches the set budget.

We make use of a trivially modified CCFG widget *à la* Ganty and Majumdar to implement transitions justified by Rule (R'-2). We will first define the rules of the CCFG widget and how it is activated.

Let us define a few abbreviations. Let $s_b^{\text{budget}} = [p_{\text{budget}}^{\text{CCFG}} \mapsto [\bullet]]$ and for each $N \in \mathcal{N}^{\text{com}}$ let $s_N = [p_N^{\text{CCFG}} \mapsto [\bullet]]$.

For each sequence of non-commutative non-terminals and cache characters $X_k \ell_k \dots X_1 \ell_1$ where $k \leq K$ we introduce the following complex rules:

* $((p_{A, (+) \Xi}^C, s_{sim}), (p_{B, (+) \Xi}^{\text{CCFG}}, \emptyset, s_{\text{CCFG}+ \oplus O}))$,
 * $((p_{A, (+) \Xi}^C, s_{sim}), (p_{B, (-) \Xi}^{\text{CCFG}}, \emptyset, s_{\text{CCFG} \oplus O}))$,
 * $((p_{A, (-) \Xi}^C, s_{sim}), (p_{B, (-) \Xi}^{\text{CCFG}}, \emptyset, s_{\text{CCFG} \oplus O}))$ and
 * $((p_{A, \ell \Xi}^C, s_{sim}), (p_{B, \ell \Xi}^{\text{CCFG}}, \emptyset, s_{\text{CCFG} \oplus O}))$ for each $\ell \in \{A_i^{\text{cov}} : i \in \langle n \rangle\}$

where $O = s_{|\mathcal{N}|}^{\text{budget}} \oplus s_C$. Each of rules activates the simulation of the CCFG widget. The topmost cache character is set according to whether the cache is a *TermCache* (+), a potential *MixedCache* (-) or exposing non-terminal A_i^{cov} in a *MixedCache*. Further the $s_{\text{CCFG}+}$ “mode” will enforce that the CCFG widget computes a *TermCache* while the s_{CCFG} “mode” allows a *MixedCache* and the exposure of a non-terminal. Further a token is placed in place p_C^{CCFG} and the budget place is initialised with $|\mathcal{N}|$ \bullet -tokens. The CCFG widget maintains the invariant that the number of tokens in the set of places $\bigcup_{N \in \mathcal{N}} p_N^{\text{CCFG}}$ plus the contents of p^{budget} equals $|\mathcal{N}| + 1$.

To exit the simulation of the CCFG widget we add the rules:

* $((p_{B, (+) \Xi}^{\text{CCFG}}, I), (p_{B, (+) \Xi}^C, \emptyset, s_{sim}))$,
 * $((p_{B, (-) \Xi}^{\text{CCFG}}, I), (p_{B, (-) \Xi}^C, \emptyset, s_{sim}))$ and
 * $((p_{B, \ell \Xi}^{\text{CCFG}}, I), (p_{B, \ell \Xi}^C, \emptyset, s_{sim}))$ for each $\ell \in \{A_i^{\text{cov}} : i \in \langle n \rangle\}$

where $I = s_{\text{CCFG}} \oplus s_{|\mathcal{N}|+1}^{\text{budget}}$. Since we require that p^{budget} contains $|\mathcal{N}| + 1$ \bullet -tokens the invariant tells us that the set of places $\bigcup_{N \in \mathcal{N}} p_N^{\text{CCFG}}$ must be empty when one of the above rules is enabled.

We turn to how the CCFG widget implements \mathcal{G} 's commutative rules. For each \mathcal{G} -rule r which involves only commutative non-terminals we do a case analysis on r :

(I) $r = X \rightarrow YZ$ and X, Y, Z commutative

We add the complex rules

- $((p_{B,(+)}^{CCFG} \Xi, s_{CCFG+} \oplus I), (p_{B,(+)}^{CCFG} \Xi, \emptyset, s_{CCFG+} \oplus O)),$
- $((p_{B,(-)}^{CCFG} \Xi, s_{CCFG} \oplus I), (p_{B,(-)}^{CCFG} \Xi, \emptyset, s_{CCFG} \oplus O)),$ and
- $((p_{B,\ell}^{CCFG} \Xi, s_{CCFG} \oplus I), (p_{B,\ell}^{CCFG} \Xi, \emptyset, s_{CCFG} \oplus O))$ for each $\ell \in \{A_i^{\text{cov}} : i \in \langle n \rangle\}$ where $I = s_X \oplus s_2^{\text{budget}}$ and $O = s_Y \oplus s_Z \oplus s_1^{\text{budget}}$. Further if $W \in \{Y, Z\}$ and $W = A_i^{\text{cov}}$ then we add the complex rule $((p_{B,(-)}^{CCFG} \Xi, s_{CCFG} \oplus I), (p_{B,W}^{CCFG} \Xi, \emptyset, s_{CCFG} \oplus O))$. We notice that a non-terminal can only be non-deterministically exposed in s_{CCFG} -“mode” and not changed after it has been set.

(II) $r = X \rightarrow e, e \in \Sigma^{\text{com}} \cup \{\epsilon\}$

We add the complex rules

- (i) $((p_{B,(+)}^{CCFG} \Xi, s_{CCFG+} \oplus I), (p_{B,(+)}^{CCFG} \Xi, c, s_{CCFG+} \oplus O)),$
- (ii) $((p_{B,(-)}^{CCFG} \Xi, s_{CCFG} \oplus I), (p_{B,(-)}^{CCFG} \Xi, c, s_{CCFG} \oplus O))$ and
- (iii) $((p_{B,\ell}^{CCFG} \Xi, s_{CCFG} \oplus I), (p_{B,\ell}^{CCFG} \Xi, c, s_{CCFG} \oplus O))$ for each $\ell \in \{A_i^{\text{cov}} : i \in \langle n \rangle\}$ where $I = s_X, O = s_1^{\text{budget}}$ and $c = [p_{k+1,e}^I \mapsto [\bullet]]$.

(III) For each $X \in \mathcal{N}^{\text{com}}$

We add the simple rule: $(s_{CCFG} \oplus I, s_{CCFG} \oplus O)$ and where $I = s_X$ and $O = s_1^{\text{budget}}$.

The CCFG widget defined above is essentially the same as Ganty and Majumdar's in [22] with one difference: our CCFG widget injects for each terminal symbol $e \in \Sigma^{\text{com}}$ a token of colour $p_{k,e}^I$ into the unique complex token located in some $p_{B,\ell}^{CCFG}$ instead of placing a token into a designated place p_e .

Further our CCFG widget can be thought of as implementing two CCFGs derived from \mathcal{G} 's rules. One, indicated by the “mode” s_{CCFG+} , implements the CCFG induced by \mathcal{G} 's commutative rules; the other, indicated by the “mode” s_{CCFG} , allows: (a) \mathcal{G} to produce partial words using rules introduced by (III), which can be thought of allowing a non-terminal X to rewrite to a terminal \bar{X} that is ignored; and (b) a non-terminal A_i^{cov} may change the location of the unique complex token m from some place $p_{B,(-)}^{CCFG} \Xi$ to $p_{B,A_i^{\text{cov}}}^{CCFG} \Xi$ which reflects the representation of the topmost cache's character for the process represented by m . This concludes the description of the implementation of Rule (R'-2).

• **Rule (R'-3):** $\Pi = \Pi', \Gamma = ([msg] \oplus q)^c, \Gamma'$ and $\gamma = (c ? msg) \gamma'$.

Let $\$ = c ? msg$. For each sequence of non-terminals and cache characters $\$ \ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K+1$ we introduce a complex rule $((p_{\$, \ell_{k+1}}^c \Xi, I), (p_{\$, \ell_{k+1}}^c \Xi, \emptyset, O))$ where $I = [p_{c, msg}^s \mapsto [\bullet]] \oplus s_{sim}$ and $O = s_{sim}$ that moves a complex token from a place encoding $\$ \ell_{k+1} \Xi$ to a place representing $\epsilon \ell_{k+1} \Xi$ while removing a \bullet -token from the simple place representing messages msg in channel c .

• **Rule (R'-4):** $\Pi = \Pi', \Gamma, ([msg] \oplus q)^c = \Gamma'$ and $\gamma = (c ! msg) \gamma'$.

Let $\$ = c ! msg$. For each sequence of non-terminals and cache characters $\$ \ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K+1$

we introduce a complex rule $((p_{\$, \ell_{k+1}}^c \Xi, I), (p_{\$, \ell_{k+1}}^c \Xi, \emptyset, O))$ where $I = s_{sim}$ and $O = [p_{c, msg}^s \mapsto [\bullet]] \oplus s_{sim}$ that moves a complex token from a place encoding $\$ \ell_{k+1} \Xi$ to a place representing $\epsilon \ell_{k+1} \Xi$ while adding a \bullet -token from the simple place representing messages msg in channel c .

• **Rule (R'-5):** $\Pi \parallel X = \Pi', \Gamma = \Gamma'$ and $\gamma = \nu X \gamma'$

Let $\$ = \nu X$. For each sequence of non-terminals and cache characters $\$ \ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K+1$ we introduce a complex rule $((p_{\$, \ell_{k+1}}^c \Xi, I), (p_{\$, \ell_{k+1}}^c \Xi, \emptyset, O))$ where $I = s_{sim}$ and $O = [p_{\nu X}^s \mapsto [\bullet]] \oplus s_{sim}$ that moves a complex token from a place encoding $\$ \ell_{k+1} \Xi$ to a place representing $\epsilon \ell_{k+1} \Xi$ while adding a \bullet -token to the simple “spawning” place $p_{\nu X}^s$. Additionally, for every $N \in \mathcal{N}$ we introduce a simple rule $r_N = ([p_{\nu N}^s \mapsto [\bullet]] \oplus s_{sim}, [p_{N,+}^s \mapsto [\emptyset]] \oplus s_{sim})$ that removes a \bullet -token from $p_{\nu N}^s$ and adds the empty complex token \emptyset to a complex place representing $N+$. For reference we will call the rule r_N a weak spawn rule for the non-terminal N .

• **Rule (R'-6):** $\Pi' = \Pi \oplus \Pi(M), \Gamma' = \Gamma \oplus \Gamma(M), \gamma = M \gamma'$ and $M \in \text{TermCache}$.

For each sequence of non-terminals and cache characters $\epsilon (+) X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K+1$ we introduce a transfer rule $((p_{\epsilon, +}^c X_k \ell_k \cdots X_1 \ell_1, s_{sim}), (p_{X_k, \ell_k \cdots X_1 \ell_1}^c, P, s_{sim}))$ where $P = \{p_{k+1,e}^I \mid e \in \Sigma^{\text{com}}\}$ if $k \leq K$ and $P = \emptyset$ otherwise. This rule moves a complex token from a place encoding $\epsilon, +, \Xi$ to a place representing Ξ while it simulates the immediate despatch of the commutative concurrency actions, send $c ! msg$ and spawn νX , that are present in the top-level cache M which is in TermCache since its character is $+$.

• **Rule (R'-7):** $\Pi' = \Pi \oplus \Pi(M), \Gamma' = \Gamma \oplus \Gamma(M), \gamma = M \gamma_0, \gamma' = M' \gamma_0, M' = M \upharpoonright (\mathcal{N}^{\text{com}} \cup \{A_i^{\text{cov}} : i \in \langle n \rangle\})$ and $M \in \text{MixedCache}$

For each sequence of non-terminals and cache characters $\epsilon \ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K+1, \ell_{k+1} \in \{-\} \cup \{A_i^{\text{cov}} : i \in \langle n \rangle\}$ where $k \leq K$ we introduce a transfer rule $((p_{\epsilon, \ell_{k+1}}^c \Xi, s_{sim}), (p_{\epsilon, \ell_{k+1}}^c \Xi, P, s_{sim}))$ where $P = \{p_{k+1,e}^I \mid e \in \Sigma^{\text{com}}\}$. This rule moves a takes a complex token from a place encoding $\epsilon, +, \Xi$ and places it back while it simulates the immediate despatch of the commutative concurrency actions, send $c ! msg$ and spawn νX , that are present in the top-level cache M which is in MixedCache since its character is $\ell_{k+1} \in \{-\} \cup \{A_i^{\text{cov}} : i \in \langle n \rangle\}$.

The coverability query $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, A_1^{\text{cov}} \parallel \cdots \parallel A_n^{\text{cov}} \triangleleft \Gamma^{\text{cov}})$ is implemented by a family of widgets W_1, \dots, W_n where for each i, W_i is a disjunction of the non-emptiness test of complex places of the shapes $p_{\epsilon, A_i^{\text{cov}}}^c \Xi$ and $p_{A_i^{\text{cov}}, \ell}^c \Xi$, as ℓ ranges over $\widehat{\mathcal{L}} := \{A_i^{\text{cov}} : i \in \langle n \rangle\} \cup \{+, -\}$, and Ξ and Ξ' range over $(\mathcal{N}^{\text{com}} \cdot \widehat{\mathcal{L}})^*$ of the appropriate lengths. A widget signals that the non-emptiness test is satisfied by placing a \bullet -token into the simple place $p_{A_i}^{\text{cov}}$. The intention is that we can use a coverability query for \mathcal{N} asking for at least one token in each of $p_{A_1}^{\text{cov}}, \dots, p_{A_n}^{\text{cov}}$ to implement the coverability query $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, A_1^{\text{cov}} \parallel \cdots \parallel A_n^{\text{cov}} \triangleleft \Gamma^{\text{cov}})$.

Formally, there is a simple rule (s_{sim}, s_{Query}) that non-deterministically terminates the simulation of \mathcal{G} and activates the processing of the coverability query. Then for each A_i^{cov} where $i \in \langle n \rangle$, we implement the widget W_i by the following complex rules: for all non-commutative non-terminals and cache characters $X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K$, and $\$ \in \Sigma \cup \{\epsilon\} \cup \mathcal{N}$ we introduce complex rules $((p_{A_i^{cov}, \ell_{k+1} \Xi}^C, s_{Query}), (p_{\epsilon, (-) \Xi}^C, \emptyset, s_{Query} \oplus O))$ and $((p_{\epsilon, A_i^{cov} \Xi}^C, s_{Query}), (p_{\epsilon, (-) \Xi}^C, \emptyset, s_{Query} \oplus O))$ where $O = [p_{A_i^{cov}}^C \mapsto [\bullet]]$.

Let us briefly inspect the size of \mathcal{N} and let us assume that all $n, K, |\mathcal{R}|, |\mathcal{N}|, |\Sigma| > 0$. It is clear that there exists constants c, c' and c'' such that \mathcal{N} has no more than $c \cdot n \cdot |\mathcal{R}| \cdot |\mathcal{P}^C|^2$ complex rules, \mathcal{N} has no more than $c' \cdot |\mathcal{R}| \cdot |\mathcal{N}|$ simple rules and \mathcal{N} has no more than $c'' \cdot |\mathcal{P}^C|^2$ transfer rules. Hence there exists a constant c''' such that \mathcal{N} has no more than $c''' \cdot n \cdot |\mathcal{R}| \cdot |\mathcal{N}| \cdot |\mathcal{P}^C|^2$ rules. Further there exists constants d, d', d'', d''' such that \mathcal{N} has no more than $d \cdot n \cdot |\Sigma| \cdot |\mathcal{N}|$ simple places, \mathcal{N} has no more than $d' \cdot K \cdot |\Sigma|$ colours, and \mathcal{N} has no more than $d'' \cdot |\mathcal{N}|^{d'''} \cdot K \cdot |\Sigma|^{d'''} \cdot K$ complex places. It is thus easy to see that \mathcal{N} can be computed from \mathcal{G} in EXPTIME.

We will now prove that it checking whether $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, A_1^{cov} \parallel \cdots \parallel A_n^{cov} \triangleleft \Gamma^{cov})$ is a yes-instance of simple coverability reduces to a coverability check on \mathcal{N} .

In order to clarify which model induces a transition we will write $\rightarrow_{\mathcal{G}}$ for $\rightarrow_{con'}$ in the following. We label the transition system $(Config^{APCPS}, \rightarrow_{\mathcal{G}})$ in the following way: if $\Pi \triangleleft \Gamma, \Pi' \triangleleft \Gamma' \in Config^{APCPS}$ such that $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ then the labelled version has the transition $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$. Next, let us define the transition system $(\mathcal{P}[Config], \rightarrow_{\mathcal{P}[\mathcal{N}]})$ which is a weak co-universal power lifting of the transition system $(Config, \rightarrow_{\mathcal{N}})$. Suppose $S, S' \subseteq Config$ then $S \rightarrow_{\mathcal{P}[\mathcal{N}]} S'$ just if for all $s' \in S'$ there exists an $s \in S$ such that there are number of weak spawn rules (c.f. \bullet) r_1, \dots, r_m where $m \geq 0$ such that $s \xrightarrow{r_1}_{\mathcal{N}} s_1 \xrightarrow{r_2}_{\mathcal{N}} \cdots \xrightarrow{r_m}_{\mathcal{N}} s_m$ and $s_m \rightarrow_{\mathcal{N}} s'$. We label $(\mathcal{P}[Config], \rightarrow_{\mathcal{P}[\mathcal{N}]})$ in the following way: suppose $S, S' \subseteq Config$ such that $S \rightarrow_{\mathcal{P}[\mathcal{N}]} S'$; if there exists $\Pi \triangleleft \Gamma \in Config^{APCPS}$ such that $S' = \mathcal{F}(\Pi \triangleleft \Gamma)$ then we label the transition by $S \xrightarrow{\Pi \triangleleft \Gamma}_{\mathcal{P}[\mathcal{N}]} S'$; otherwise we label the transition by $S \xrightarrow{\epsilon}_{\mathcal{P}[\mathcal{N}]} S'$.

We will now prove that R is a weak bisimulation between the labelled versions of $(Config^{APCPS}, \rightarrow_{\mathcal{G}})$ and $(\mathcal{P}[Config], \rightarrow_{\mathcal{P}[\mathcal{N}]})$.

Let us first prove R is a weak simulation. Suppose $(\gamma \parallel \Pi \triangleleft \Gamma, \mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma)) \in R$ and $\gamma \parallel \Pi \triangleleft \Gamma \xrightarrow{\Pi_0 \triangleleft \Gamma_0}_{\mathcal{G}} \gamma' \parallel \Pi' \triangleleft \Gamma'$ then clearly $\gamma \parallel \Pi \triangleleft \Gamma \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}_{\mathcal{G}} \gamma' \parallel \Pi' \triangleleft \Gamma'$.

We want to show that $\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$. So let $s' \in \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ then by definition $s' = s'_0 \oplus (\mathcal{F}^r(\Gamma') \oplus s_{sim})$ for some $s'_0 \in \mathcal{F}^n(\gamma' \parallel \Pi')$. We can further decompose s'_0 and obtain $s'_0 = s'_1 \oplus s'_2$ where $s'_1 \in \mathcal{F}^n(\gamma')$ and $s'_2 \in \mathcal{F}^n(\Pi')$.

Let us perform a case analysis on the rule that justifies the transition $\gamma \parallel \Pi \triangleleft \Gamma \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}_{\mathcal{G}} \gamma' \parallel \Pi' \triangleleft \Gamma'$:

- **Rule (R'-1):** $\Pi = \Pi', \Gamma = \Gamma', \gamma = A M X_k M_k \cdots X_1 M_1$, and $\gamma' = \delta M X_k M_k \cdots X_1 M_1$ for some \mathcal{G} rule $A \rightarrow \delta$.

We can see that $s'_1 = [p_{\delta, \ell_{k+1} \Xi}^C \mapsto [\mathcal{F}^i(M_1, \dots, M_{k+1})]]$ where ℓ_i is a character of M_i for each $i \in \langle k+1 \rangle$, and $\bar{\delta}$ depends on the form of δ : $\bar{\delta} = B (+) C$ if $\delta = B C$, $k < K$ and C non-commutative; and $\bar{\delta} = a$ if $\delta = a$ and $a \in \Sigma \cup \{\epsilon\}$. Note that it is now obvious that it is impossible that γ' is represented in a simple place.

Let $s = (s_1 \oplus s'_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim})$ we write $s_1 := [p_{A, \ell_{k+1} \Xi}^C \mapsto [\mathcal{F}^i(M_1, \dots, M_{k+1})]]$. Clearly $s_1 \in \mathcal{F}^n(A M X_k M_k \cdots X_1 M_1) = \mathcal{F}^n(\gamma)$ and thus $s \in \mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma)$.

We know that \mathcal{N} has a complex rule $((p_{A, \ell_{k+1} \Xi}^C, s_{sim}), (p_{\delta, \ell_{k+1} \Xi}^C, \emptyset, s_{sim}))$. Thus we know we can make the transition $s \rightarrow_{\mathcal{N}} (s'_1 \oplus s'_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim})$ and clearly $s' = (s'_1 \oplus s'_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim})$.

Since $s \in \mathcal{F}(\gamma \parallel \Pi' \triangleleft \Gamma')$ was arbitrary we can conclude that in fact $\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \rightarrow_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ and thus clearly

$$\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$$

which is what we wanted to prove.

- **Rule (R'-2):** $\Pi = \Pi', \Gamma = \Gamma', \gamma = A M X_k M_k \cdots X_1 M_1$, $\gamma' = \delta M' X_k M_k \cdots X_1 M_1$ for some \mathcal{G} rule $A \rightarrow B C$, C commutative, $C \rightarrow^* w$, and $M' = M \oplus \mathbb{M}(w)$.

In this case we can assume that $s'_1 = [p_{B, \ell' \Xi}^C \mapsto [\mathcal{F}^i(M_1, \dots, M_k, M')]]$ where ℓ' is a character for $M' = M \oplus \mathbb{M}(w)$.

Let us define cache characters $\ell, \bar{\ell}$ and $\bar{\ell}'$ by a case analysis on ℓ' and $\mathbb{M}(w)$:

- (C1) Case: $\ell' = +$: Define $\ell = \bar{\ell} = \bar{\ell}' = +$.
- (C2) Case: $\ell' = -$ and $M \in MixedCache$: Define $\ell = \bar{\ell} = \bar{\ell}' = -$.
- (C3) Case: $\ell' = -$ and $M \in TermCache$: Define $\ell = +$ and $\bar{\ell} = \bar{\ell}' = -$.
- (C4) Case: $\ell' \in \{A_i^{cov} : i \in \langle n \rangle\}$, $M \in MixedCache$ and $\mathbb{M}(w)(\ell') > 0$: Define $\ell = \bar{\ell} = -$ and $\bar{\ell}' = \ell'$.
- (C5) Case: $\ell' \in \{A_i^{cov} : i \in \langle n \rangle\}$, $M \in TermCache$ and $\mathbb{M}(w)(\ell') > 0$: Define $\ell = +$ and $\bar{\ell} = -$ and $\bar{\ell}' = \ell'$.
- (C6) Case: $\ell' \in \{A_i^{cov} : i \in \langle n \rangle\}$ and $\mathbb{M}(w)(\ell') = 0$: Define $\ell = \bar{\ell} = \bar{\ell}' = \ell'$.

Let us show that the above choices ensure that ℓ is a character for M . If $M \in TermCache$ then either $M' = M \oplus \mathbb{M}(w) \in TermCache$ and so $\ell' = +$ and hence $\ell = +$ and thus ℓ is a character for M . Otherwise $\ell' \in \{-\} \cup \{A_i^{cov} : i \in \langle n \rangle\}$ and so since $M \in TermCache$ by definition we have $\ell = +$ and hence ℓ is a character for M . If $M \in MixedCache$ then clearly $M \oplus \mathbb{M}(w) \in MixedCache$ and so either $\ell' = -$ or $\ell' \in \{A_i^{cov} : i \in \langle n \rangle\}$. So if $\ell' = -$ or $\mathbb{M}(w)(\ell') > 0$ then $\ell = -$ which is a character M . Otherwise, $\ell' \in \{A_i^{cov} : i \in \langle n \rangle\}$ and $\mathbb{M}(w)(\ell') = 0$ then $\ell = \ell'$ and since ℓ' is not a character for $\mathbb{M}(w)$ but for M' it must be that case that $\ell = \ell'$ is character for M . Hence we can conclude that in all cases ℓ is a character for M .

Define $s^{s'}$ by $s^{s'} = s'_1 \oplus s'_2 \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim})$ where $s'_1 = [p_{A, \ell \Xi}^C \mapsto [\mathcal{F}^i(M_1, \dots, M_k, M)]]$. It is clear that $s^{s'} \in \mathcal{F}^n(\gamma)$

and hence $s^{s'} \in \mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma)$.

Our definitions of ℓ and $\bar{\ell}$ ensure that there is a complex rule of the form $\mathbf{r}(1) = ((p_{A,\ell}^C, s_{sim}), (p_{B,\bar{\ell}}^{CCFG}, \emptyset, s_{CCFG+?} \oplus O))$ which is enabled at $s^{s'}(1)$ to active a CCFG widget computation where $CCFG+? = CCFG +$ if $\bar{\ell} = +$; and $CCFG+? = CCFG$ otherwise; and $O = s_{|\mathcal{N}|}^{budget} \oplus [p_C^{CCFG} \mapsto [\bullet]]$.

Hence we can make a transition $s^{s'}(1) \xrightarrow{\mathbf{r}(1)}_{\mathcal{N}} s^{s'}(2)$ where $s^{s'}(2) = (s_1^{s'}(2) \oplus s_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{CCFG+?} \oplus s_{|\mathcal{N}|}^{budget} \oplus [p_C^{CCFG} \mapsto [\bullet]])$ and $s^{s'}(2) = [p_{B,\bar{\ell}}^{CCFG} \mapsto [\mathcal{F}^r(M_1, \dots, M_{k+1})]]$.

We appeal to [22] that the CCFG widget allows us to simulate the computation $C \rightarrow^* w$ in L steps, where L only depends on the derivation $C \rightarrow^* w$ and not on $s^{s'}(2)$, so that we get $s^{s'}(1+1) \xrightarrow{\mathbf{r}(2)}_{\mathcal{N}} s^{s'}(1+2) \dots \xrightarrow{\mathbf{r}(1+L)}_{\mathcal{N}} s^{s'}(1+L)$ where for all $2 \leq i \leq L+1$ the rule $\mathbf{r}(i)$ is introduced by cases (I)–(III) above; and for each terminal $e \in \Sigma^{\text{com}}$ occurring in w we inject a $p_{k+1,e}^i$ -coloured token into $\mathcal{F}^r(M_1, \dots, M_{k+1})$, i.e. if $M_0 \leq_{\mathbb{M}} \mathbb{M}(w)$ then injecting a $p_{k+1,e}^i$ -coloured token $\mathcal{F}^r(M_1, \dots, M_{k+1} \oplus M_0)$ yields $\mathcal{F}^r(M_1, \dots, M_{k+1} \oplus (M_0 \oplus [e]))$, and for each $2 \leq i \leq L+1$ the configuration $s^{s'}(i)$ has $s_{CCFG+?}$ as a submarking. Further, we can see that in the computation of $C \rightarrow^* w$ it is possible to expose a non-terminal ℓ_0 if $\mathbb{M}(w)(\ell_0) > 0$ and $\bar{\ell} = -$. We can thus see that if cases (C1)–(C5) applied for the definition of $\bar{\ell}$ and $\bar{\ell}'$ then we can assume that the above computation ignores all non-terminal; if case (C6) applied then $\mathbb{M}(w)(\ell') > 0$ and $\bar{\ell} = -$, so we can expose $\ell' = \bar{\ell}'$ along the above computation.

Hence we can conclude that $s^{s'}(1+L) = (s_1^{s'}(1+L) \oplus s_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{CCFG+?} \oplus s_{|\mathcal{N}|+1}^{budget})$ where $s^{s'}(1+L) = [p_{B,\bar{\ell}'}^{CCFG} \mapsto [\mathcal{F}^r(M_1, \dots, M_{k+1} \oplus (\mathbb{M}(w) \upharpoonright \Sigma^{\text{com}}))]]$.

We note that $\bar{\ell}' = \ell'$ in all cases of (C1)–(C6).

It is then the case that a complex rule of the form $((p_{B,\ell'}^{CCFG}, s_{sim}), (p_{B,\ell'}^C, \emptyset, s_{sim}))$ is enabled and we can make the transition $s^{s'}(L+1) \rightarrow_{\mathcal{N}} s^{s'}(L+2)$ where $s^{s'}(L+2) = (s_1^{s'}(L+2) \oplus s_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim})$ and $s^{s'}(L+2) = [p_{B,\ell'}^C \mapsto [\mathcal{F}^r(M_1, \dots, M_{k+1} \oplus (\mathbb{M}(w) \upharpoonright \Sigma^{\text{com}}))]]$.

Now $\mathcal{F}^r(M_1, \dots, M_{k+1} \oplus (\mathbb{M}(w) \upharpoonright \Sigma^{\text{com}})) = \mathcal{F}^r(M_1, \dots, M \oplus \mathbb{M}(w))$. Thus $s^{s'}(L+2) = s'_1$ and hence $s^{s'}(L+2) = s'$.

We now need to lift these paths to $\mathcal{P}[\mathcal{N}]$. The recipe above gives us for each $s' \in \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ a path $s^{s'}(1) \dots s^{s'}(L+2)$. We note that L is in fact independent of s' since it is only dependent on the derivation $C \rightarrow^* w$. Further $s^{s'}(1) \in \mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma)$, $s^{s'}(L+2) = s'$ and each for all $2 \leq i < L+1$ the configuration $s^{s'}(i)$ contains either submarking s_{CCFG+} or s_{CCFG} and hence $\# \Pi_0 \triangleleft \Gamma_0$ such that $s^{s'}(i) \in \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$. Let us define the following subsets of Config : for $1 \leq i \leq L+2$ let $\mathbf{S}(i) = \{s^{s'}(i) \mid s' \in \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')\}$. By definition we have for all $s^{s'}(i+1) \in \mathbf{S}(i+1)$ that $s^{s'}(i) \rightarrow_{\mathcal{N}} s^{s'}(i+1) \in \mathbf{S}(i+1)$ if $1 \leq i < L+2$ and hence $\mathbf{S}(1) \rightarrow_{\mathcal{P}[\mathcal{N}]} \mathbf{S}(2) \rightarrow_{\mathcal{P}[\mathcal{N}]} \dots \rightarrow_{\mathcal{P}[\mathcal{N}]} \mathbf{S}(L+1) \rightarrow_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ since clearly $\mathbf{S}(L+2) = \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$. Looking at the labelled version of $\mathcal{P}[\mathcal{N}]$ our reasoning above implies that For all $1 < i < L+2$ $\# \Pi_0 \triangleleft \Gamma_0$ such that $\mathbf{S}(i) = \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$ and hence

$\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \supseteq \mathbf{S}(1) \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ which is what we wanted to prove.

- **Rule (R'-3), (R'-4)**: $\gamma = \$ \gamma'$ and $\$ \in \Sigma$.

We will prove the case where the transition is justified by rule (R'-3); the case using rule (R'-4) is analogous.

In this case $\Pi = \Pi'$, $\Gamma = ([msg] \oplus q)^c, \Gamma_0$, $\Gamma' = q^c, \Gamma_0$ and $\$ = c ? msg$.

If $\gamma' = M_{k+1} X_k M_k \dots X_1 M_1$ is easy to see that $s'_1 = [p_{\epsilon, \ell_{k+1}}^c \mapsto [\mathcal{F}^r(M_1, \dots, M_{k+1})]]$ where ℓ_i is a character of M_i for each $i \in \langle k+1 \rangle$; and $\mathcal{F}^r(\Gamma) = \mathcal{F}^r(\Gamma') \oplus [p_{c, msg}^s \mapsto [\bullet]]$.

Hence let $s = (s_1 \oplus s'_2) + (\mathcal{F}^r(\Gamma) \oplus s_{sim})$ where we write $s_1 := [p_{c ? msg, \ell_{k+1}}^c \mapsto [\mathcal{F}^r(M_1, \dots, M_{k+1})]]$.

The complex rule $((p_{c ? msg, \ell_{k+1}}^c, [p_{c, msg}^s \mapsto [\bullet]]) \oplus s_{sim}), (p_{\epsilon, \ell_{k+1}}^c, \emptyset, s_{sim}))$ is then enabled at s and we can make the transition $s \rightarrow_{\mathcal{N}} (s'_1 \oplus s'_2) + (\mathcal{F}^r(\Gamma') \oplus s_{sim}) = s'$

Since $\gamma = (c ? msg) \gamma'$ we know that $s_1 \in \mathcal{F}^{\pi}((c ? msg) \gamma')$ from which we can conclude that $s_1 \oplus s'_2 \in \mathcal{F}^{\pi}(\gamma \parallel \Pi') = \mathcal{F}^{\pi}(\gamma \parallel \Pi)$ and hence $(s_1 \oplus s'_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim}) \in \mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma)$.

Since $s' \in \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ was arbitrary we can conclude that in fact $\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \rightarrow_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ and thus clearly

$$\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$$

which concludes this case.

- **Rule (R'-5)**: $\Pi' = \Pi \parallel X$, $\Gamma = \Gamma'$, and $\gamma = \nu X \gamma'$.

If $\gamma' = M_{k+1} X_k M_k \dots X_1 M_1$ is easy to see that $s'_1 = [p_{\epsilon, \ell_{k+1}}^c \mapsto [\mathcal{F}^r(M_1, \dots, M_{k+1})]]$ where ℓ_i is a character of M_i for each $i \in \langle k+1 \rangle$. Further since $s'_2 \in \mathcal{F}^{\pi}(\Pi') = \mathcal{F}^{\pi}(\Pi \parallel X)$ we know that $s'_2 = s''_2 \oplus s'_X$ where $s''_2 \in \mathcal{F}^{\pi}(\Pi)$ and $s'_X \in \mathcal{F}^{\pi}(X)$.

Thus we can define $s = (s_1 \oplus s''_2) + (\mathcal{F}^r(\Gamma) \oplus s_{sim})$ where we write $s_1 := [p_{\nu X, \ell_{k+1}}^c \mapsto [\mathcal{F}^r(M_1, \dots, M_{k+1})]]$.

Hence the complex rule

$$((p_{\nu X, \ell_{k+1}}^c, s_{sim}), (p_{\epsilon, \ell_{k+1}}^c, \emptyset, [p_{\nu X}^s \mapsto [\bullet]]) \oplus s_{sim}))$$

is enabled at s and we can make the transition $s \rightarrow_{\mathcal{N}} (s'_1 \oplus s''_2 \oplus s_X) \oplus (\mathcal{F}^r(\Gamma') \oplus s_{sim})$ where we write $s_X = [p_{\nu X}^s \mapsto [\bullet]]$.

Since $s'_X \in \mathcal{F}^{\pi}(X)$ either $s'_X = s_X$ or $s'_X = [p_{X, (+)}^c \mapsto [\emptyset]]$. In the latter case we can use a weak spawn rule to make the transition

$$(s'_1 \oplus s''_2 \oplus s_X) \oplus (\mathcal{F}^r(\Gamma') \oplus s_{sim}) \rightarrow_{\mathcal{N}} s'.$$

Since $\gamma = (\nu X) \gamma'$ we know that $s_1 \in \mathcal{F}^{\pi}(\gamma)$ from which we can conclude that $s_1 \oplus s''_2 \in \mathcal{F}^{\pi}(\gamma \parallel \Pi)$. Therefore $(s_1 \oplus s''_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim}) \in \mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma)$.

Since $s' \in \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ was arbitrary we can conclude that $\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \rightarrow_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ and thus clearly

$$\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$$

which is what we wanted to prove.

- **Rule (R'-6)–(R'-7)**: $\Pi' = \gamma' \parallel \Pi \parallel \Pi(M)$, $\Gamma' = \Gamma \oplus \Gamma(M)$, $\gamma = \epsilon M \gamma_0$ and $\gamma' = \epsilon M' \gamma_0$.

We will prove the case where the transition is justified by rule (R'-6); the case using rule (R'-7) is proved similarly.

In this case, in fact $M' = \emptyset$ and thus $\gamma' = \gamma_0$ which implies $\gamma = \epsilon M \gamma'$. If $\gamma' = X_k M_k \cdots X_1 M_1$ is easy to see that $s'_1 = [p_{X_k \ell_k \cdots X_1 \ell_1}^C \mapsto [\mathcal{F}^I(M_1, \dots, M_k)]]$ where ℓ_i is a character of M_i for each $i \in \langle k \rangle$. Further since $s'_2 \in \mathcal{F}^n(\Pi') = \mathcal{F}^n(\Pi \parallel \Pi(M))$ we decompose s'_2 into $s'_2 = s''_2 \oplus s'_{\Pi(M)}$ where $s''_2 \in \mathcal{F}^n(\Pi)$ and $s'_{\Pi(M)} \in \mathcal{F}^n(\Pi(M))$.

We also know that $M \in \text{TermCache}$, so let $\ell = +$, $s = (s_1 \oplus s''_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim})$ and $s_1 = [p_{\epsilon, + \Xi}^C \mapsto [\mathcal{F}^I(M_1, \dots, M_k, M)]]$. Thus $s_1 \in \mathcal{F}^n(M \gamma') = \mathcal{F}^n(\gamma)$ and hence $s \in \mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma)$.

The transfer rule $r = ((p_{\epsilon, + \Xi}^C, s_{sim}), (p_{\Xi}^C, P, s_{sim}))$ is then enabled at s where $P = \{p_{k+1, e}^C \mid e \in \Sigma^{\text{com}}\}$ if $k \leq K$ and $P = \emptyset$ otherwise (in which case $M = \emptyset$). Let $m = \mathcal{F}^I(M_1, \dots, M_k, M)$, $m_P = m \upharpoonright P$ and $m_{\bar{P}} = m \upharpoonright (\mathcal{P}^{\text{col}} \setminus P)$. Then we know that using r we can make the transition

$$s \rightarrow_{\mathcal{N}} (s_{\bar{P}} \oplus s''_2) \oplus (\mathcal{F}^r(\Gamma) \oplus s_{sim}) \oplus (m_P \circ \zeta^{-1}) =: s'_0$$

where $s_{\bar{P}} = [p_{\Xi}^C \mapsto [m_{\bar{P}}]]$.

It is not hard to see that $m_{\bar{P}} = \mathcal{F}^I(M_1, \dots, M_k, \emptyset) = \mathcal{F}^I(M_1, \dots, M_k)$ and hence $s_{\bar{P}} = s'_1$.

Let $\mathcal{G}^r = \{p_{c, msg}^s \mid c \in \text{Chan}, msg \in \text{Msg}\}$ and $\mathcal{G}^\nu = \{p_{\nu X}^s \mid X \in \mathcal{N}\}$. We notice that for all $c \in \text{Chan}, msg \in \text{Msg}$

$$\begin{aligned} \mathcal{F}^r(\Gamma(M \upharpoonright \{c ! msg\})) &= \mathcal{F}^r \left([msg^{M(c!msg)}]^c \right) \\ &= [p_{c, msg}^s \mapsto [\bullet^{M(c!msg)}]] \\ &= [p_{c, msg}^s \mapsto m_P(c ! msg)] \\ &= (m_P \circ \zeta^{-1}) \upharpoonright \{p_{c, msg}^s\} \end{aligned}$$

from which we conclude that

$$\begin{aligned} (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^r &= \bigoplus_{\substack{c \in \text{Chan}, \\ msg \in \text{Msg}}} \mathcal{F}^r(\Gamma(M \upharpoonright \{c ! msg\})) \\ &= \mathcal{F}^r(\Gamma(M)) \end{aligned}$$

and $\mathcal{F}^r(\Gamma') = \mathcal{F}^r(\Gamma) \oplus (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^r$.

Further for each $X \in \mathcal{N}$ let $s_X = [p_{\nu X}^s \mapsto [\bullet]]$; then it is the case that $\bigoplus_{i=1}^{M(\nu(X))} s_X \in \mathcal{F}^n(\Pi(M \upharpoonright \{\nu X\}))$ and thus $\bigoplus_{X \in \mathcal{N}} \bigoplus_{i=1}^{M(\nu(X))} s_X \in \mathcal{F}^n(\Pi(M))$. Now $\bigoplus_{i=1}^{M(\nu(X))} s_X = (m_P \circ \zeta^{-1}) \upharpoonright \{p_{\nu X}^s\}$ and thus $\bigoplus_{X \in \mathcal{N}} \bigoplus_{i=1}^{M(\nu(X))} s_X = (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\nu$. Since $(m_P \circ \zeta^{-1}) = (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^r \oplus (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\nu$ we have

$$s'_0 = (s'_1 \oplus s_2 \oplus (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\nu) \oplus (\mathcal{F}^r(\Gamma') \oplus s_{sim}).$$

Let us inspect $s'_{\Pi(M)}$. Since $s'_{\Pi(M)} \in \mathcal{F}^n(\Pi(M))$ either $s'_{\Pi(M)} = m_P \circ \zeta^{-1} \upharpoonright \mathcal{G}^\nu$ or using a number of weak spawn rules we can transition in s'_0 from $m_P \circ \zeta^{-1} \upharpoonright \mathcal{G}^\nu$ to $s'_{\Pi(M)}$. This implies (using a number of weak spawn rules after the first step) we can make the transition $s \rightarrow_{\mathcal{N}} s'_0 \rightarrow_{\mathcal{N}}^* s'$.

Since $s' \in \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ was arbitrary we can conclude that in fact $\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \rightarrow_{\mathcal{D}[\mathcal{N}]}^* \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$ and thus

clearly

$$\mathcal{F}(\gamma \parallel \Pi \triangleleft \Gamma) \xrightarrow{\gamma' \parallel \Pi' \triangleleft \Gamma'}^*_{\mathcal{D}[\mathcal{N}]} \mathcal{F}(\gamma' \parallel \Pi' \triangleleft \Gamma')$$

which is what we wanted to prove.

We can thus deduce that R is a weak simulation.

Let us first prove a lemma that will be the basis of our proof that R^{-1} is a weak simulation.

Lemma. Suppose $s(1), \dots, s(m)$ is a sequence of configurations such that (A) for each $i \in \langle m-1 \rangle$ $s(i) = s_0^i \xrightarrow{r^i(1)}_{\mathcal{N}} s_1^i \xrightarrow{r^i(2)}_{\mathcal{N}} \dots \xrightarrow{r^i(m^i-1)}_{\mathcal{N}} s_{m^i-1}^i \xrightarrow{r^i(m^i)}_{\mathcal{N}} s_{m^i}^i = s(i+1)$ where only one rule, $r^i(j^i)$ say, is not a weak spawn rule; (B) $s(1) \in \mathcal{F}(\Pi \triangleleft \Gamma)$ and $s(m) \in \mathcal{F}(\Pi' \triangleleft \Gamma')$; and (C) for all $2 \leq i < m$ there does not exist $\Pi_0 \triangleleft \Gamma_0$ such that $s(i) \in \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$. Then $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$.

Proof. It is easy to see that if a configuration $s \in \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$ for some $\Pi_0 \triangleleft \Gamma_0$ and $s \xrightarrow{r}_{\mathcal{N}} s'$ where r is a weak spawn rule then $s' \in \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$. Hence if we look at the transitions $s(1) \xrightarrow{r^1(1)}_{\mathcal{N}} s_1^1 \xrightarrow{r^1(2)}_{\mathcal{N}} \dots \xrightarrow{r^1(j^1)}_{\mathcal{N}} s_{j^1}^1$ we can conclude that for all $0 \leq l < j^1$ we have $s_l^1 \in \mathcal{F}(\Pi \triangleleft \Gamma)$. Thus we can decompose $s_{j^1-1}^1$ as $s_{j^1-1}^1 = s^\Pi \oplus \mathcal{F}^r(\Gamma) \oplus s_{sim}$ where $s^\Pi \in \mathcal{F}^n(\Pi)$.

Let us do a case analysis on which rule of the ?? justifies the introduction of $r^1(j^1)$ into \mathcal{N} 's rules.

- **Rule (R'-2).**

There are many candidate rules $r^1(j^1)$ at first sight. However, we know that $s_{j^1-1}^1$ does not have s_{CCFG} or s_{CCFG+} as a submarking. Hence we must have $r^1(j^1) = ((p_{A, \ell_{k+1} \Xi}^C, s_{sim}), (p_{B, \bar{\ell} \Xi}^{CCFG}, \emptyset, s_{CCFG+?} \oplus O))$ for some \mathcal{G} rule $A \rightarrow BC$ where C is commutative and $\bar{\ell} \in \{+, -\}$ and $CCFG+? \in \{CCFG+, CCFG\}$ if $\ell_{k+1} = +$; $\bar{\ell} = \ell_{k+1}$, and $CCFG+? = CCFG$ otherwise; and $O = s_{|\mathcal{N}|}^{\text{budget}} \oplus [p_{C, \bar{\ell} \Xi}^{CCFG} \mapsto [\bullet]]$.

Thus with $s_{j^1}^1$ a computation of the CCFG starts and until for some i, j the rule $r^i(j)$ exists the CCFG computation, i.e. $r^i(j) = ((p_{B, \bar{\ell} \Xi}^{CCFG}, s_{CCFG+?} \oplus s_{|\mathcal{N}|+1}^{\text{budget}}), (p_{B, \bar{\ell} \Xi}^C, \emptyset, s_{sim}))$ where $\bar{\ell}' \in \{-\} \cup \{A_i^{\text{cov}} \mid i \in \langle n \rangle \mathbb{M}(w)(A_i^{\text{cov}}) > 0\}$ if $\bar{\ell} = -$ and $\bar{\ell}' = \bar{\ell}$ otherwise. Along this path between the configurations $s_{j^1}^1$ and s_j^i the submarking s_{sim} cannot appear and hence no weak spawning rule can apply. Thus we know that $m^1 = j^1$, $j = 1$ and for $1 < i' < i$ it is the case that $m^{i'} = 1$. We know that $s^\Pi \in \mathcal{F}^n(\Pi)$ and hence $\Pi = A M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0$ and $[p_{A, \ell_{k+1} \Xi}^C \mapsto [m]] \in \mathcal{F}^n(A M_{k+1} X_k M_k \cdots X_1 M_1)$ where $m = \mathcal{F}^I(M_1, \dots, M_{k+1})$. Further it is clear that $m \in p_{A, \ell_{k+1} \Xi}^C$ in $s_{j^1-1}^1$. Hence the CCFG widget guarantees [22] that $s_j^i = (s^\Pi \oplus [p_{A, \ell_{k+1} \Xi}^C \mapsto [m]] \oplus [p_{B, \bar{\ell} \Xi}^C \mapsto [m']]) \oplus \mathcal{F}^r(\Gamma) \oplus s_{sim}$ where $m' = \mathcal{F}^I(M_1, \dots, M_{k+1} \oplus \mathbb{M}(w))$ for some w such that $C \rightarrow^* w$.

Further the CCFG widget guarantees that if $\bar{\ell}' = +$ then $M_{k+1} \oplus \mathbb{M}(w) \in \text{TermCache}$ and $M_{k+1} \oplus \mathbb{M}(w) \in \text{MixedCache}$ otherwise. Hence $[p_{B, \bar{\ell} \Xi}^C \mapsto [m]] \in \mathcal{F}^n(B, (M_{k+1} \oplus \mathbb{M}(w)) X_k M_k \cdots X_1 M_1)$ and we can deduce that

$s^\Pi \ominus [p_{A,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{B,\ell'}^c \mapsto [m']] \in \mathcal{F}^\Pi(B(M_{k+1} \oplus \mathbb{M}(w)) X_k M_k \cdots X_1 M_1 \parallel \Pi_0)$. It is thus the case that $s_{j_1}^1 \in \mathcal{F}^\Pi(B(M_{k+1} \oplus \mathbb{M}(w)) X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \triangleleft \Gamma)$ which implies (a) $m^1 = 1$, (b) $m = i$ and thus (c) $\Pi' \triangleleft \Gamma' = B(M_{k+1} \oplus \mathbb{M}(w)) X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \triangleleft \Gamma$. We can easily check that $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ and hence $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ which is what we wanted to prove.

- **Rule (R'-1).**

We can assume that $\mathbf{r}^1(j^1) = ((p_{A,\ell_{k+1}}^c \ominus, s_{sim}), (p_{\delta,\ell_{k+1}}^c \ominus, \emptyset, s_{sim}))$ for some non-terminals and cache characters $A\ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ and \mathcal{G} rule $A \rightarrow \delta$ where $k \leq K+1$ and δ depends on δ and k : $\delta = B, (+)C$ if $\delta = BC, k < K$; and $\delta = a$ if $\delta = a$ with $a \in \Sigma \cup \{\epsilon\}$.

Hence we can deduce there exists a complex token m located at place $p_{A,\ell_{k+1}}^c$ in $s_{j_1-1}^1$ and $s_{j_1}^1 = (s^\Pi \ominus [p_{A,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\delta,\ell_{k+1}}^c \mapsto [m]]) \oplus \mathcal{F}^\Pi(\Gamma) \oplus s_{sim}$.

We know $s^\Pi \in \mathcal{F}^\Pi(\Pi)$ and hence $\Pi = A M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0$ and $[p_{A,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(A M_{k+1} X_k M_k \cdots X_1 M_1)$. Then clearly $[p_{\delta,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\delta M_{k+1} X_k M_k \cdots X_1 M_1)$ and we can deduce that $s^\Pi \ominus [p_{A,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\delta,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\delta M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0)$.

It is thus easy to see that $s_{j_1}^1 \in \mathcal{F}^\Pi(\delta M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \triangleleft \Gamma)$. From our assumptions above this implies (a) $m^1 = j^1$, (b) $m = 1$ and thus (c) $\Pi' \triangleleft \Gamma' = \delta M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \triangleleft \Gamma$. It is also easy to see that $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ and hence $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ which is what we wanted to prove.

- **Rules (R'-3),(R'-4).**

We will prove the case that the introduction of $\mathbf{r}^1(j^1)$ is justified by rule (R'-4). The case of (R'-3) are proved similarly.

We can thus assume that $\mathbf{r}^1(j^1) = ((p_{c!msg,\ell_{k+1}}^c \ominus, s_{sim}), (p_{\epsilon,\ell_{k+1}}^c \ominus, \emptyset, [p_{c!msg}^c \mapsto [\bullet]] \oplus s_{sim}))$ for some non-terminals and cache characters $\ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K+1$. Hence there exists a complex token m located at place $p_{c!msg,\ell_{k+1}}^c$ and $s_{j_1}^1 = (s^\Pi \ominus [p_{c!msg,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m]]) \oplus (\mathcal{F}^\Pi(\Gamma) \oplus [p_{c!msg}^c \mapsto [\bullet]]) \oplus s_{sim}$.

Since $s^\Pi \in \mathcal{F}^\Pi(\Pi)$ we can deduce that $\Pi = c!msg M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0$ and $[p_{c!msg,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(c!msg M_{k+1} X_k M_k \cdots X_1 M_1)$. Then clearly $[p_{\epsilon,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1)$ and we can deduce that $s^\Pi \ominus [p_{c!msg,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0)$. Further it is easy to see that $\mathcal{F}^\Pi(\Gamma) \oplus [p_{c!msg}^c \mapsto [\bullet]] = \mathcal{F}^\Pi(\Gamma \oplus [msg]^c)$. We can then conclude that $s_{j_1}^1 \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \triangleleft \Gamma \oplus [msg]^c)$. From our assumptions above this implies (a) $m^1 = j^1$, (b) $m = 1$ and thus (c) $\Pi' \triangleleft \Gamma' = \epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \triangleleft \Gamma \oplus [msg]^c$. It is also easy to see that $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ and hence $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ which is what we wanted to prove.

- **Rule (R'-5).**

Since by definition $\mathbf{r}^1(j^1)$ is not a weak spawn rule, we can assume that $\mathbf{r}^1(j^1) = ((p_{\nu X,\ell_{k+1}}^c \ominus, s_{sim}), (p_{\epsilon,\ell_{k+1}}^c \ominus, \emptyset, s_X \oplus s_{sim}))$ for $X \in \mathcal{N}$ and some non-terminals and

cache characters $\ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K+1$ and $s_X = [p_{\nu X}^c \mapsto [\bullet]]$. Hence there exists a complex token m located at place $p_{\nu X,\ell_{k+1}}^c$ and $s_{j_1}^1 = (s^\Pi \ominus [p_{\nu X,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m]]) \oplus (\mathcal{F}^\Pi(\Gamma) \oplus [p_{\nu X}^c \mapsto [\bullet]]) \oplus s_{sim}$.

Since $s^\Pi \in \mathcal{F}^\Pi(\Pi)$ we can deduce that $\Pi = \nu X M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0$ and $[p_{\nu X,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\nu X M_{k+1} X_k M_k \cdots X_1 M_1)$. Then clearly $[p_{\epsilon,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1)$ and we can deduce that $s' := s^\Pi \ominus [p_{\nu X,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0)$. Clearly $s_X \in \mathcal{F}^\Pi(X)$ and thus we can deduce that $s' \oplus s_X \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \parallel X)$.

We can then conclude that $s_{j_1}^1 \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \parallel X \triangleleft \Gamma)$. From our assumptions above this implies (a) $m^1 = j^1$, (b) $m = 1$ and thus (c) $\Pi' \triangleleft \Gamma' = \epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \parallel X \triangleleft \Gamma$. It is also easy to see that $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ and hence $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ which is what we wanted to prove.

- **Rule (R'-6)-(R'-7).**

We will give a proof of the case that the introduction of $\mathbf{r}^1(j^1)$ is justified by rule (R'-7). The proof in the case that $\mathbf{r}^1(j^1)$ introduced to implement rule (R'-6) is analogous.

We can thus assume that $\mathbf{r}^1(j^1) = ((p_{\epsilon,\ell_{k+1}}^c \ominus, s_{sim}), (p_{\epsilon,\ell_{k+1}}^c \ominus, P, s_{sim}))$ for some non-terminals and cache characters $\ell_{k+1} X_k \ell_k \cdots X_1 \ell_1$ where $k \leq K, P = \{p_{k+1,e}^i \mid e \in \Sigma^{\text{com}}\}$ and $\ell_{k+1} \in \{-\} \cup \{A_i^{\text{cov}} : i \in \langle n \rangle\}$.

Hence there exists a complex token m located at place $p_{\epsilon,\ell_{k+1}}^c$ and $s_{j_1}^1 = (s^\Pi \ominus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m_{\overline{P}}]]) \oplus (\mathcal{F}^\Pi(\Gamma) \oplus (m_P \circ \zeta^{-1})) \oplus s_{sim}$ where $m_P = m \upharpoonright P$ and $m_{\overline{P}} = m \upharpoonright (P^{\text{col}} \setminus P)$.

Since $s^\Pi \in \mathcal{F}^\Pi(\Pi)$ we can deduce that $\Pi = \epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0$ and $[p_{\epsilon,\ell_{k+1}}^c \mapsto [m]] \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1)$. We further know that $m = \mathcal{F}^i(M_1, \dots, M_k, M_{k+1})$ and $M_{k+1} \in \text{MixedCache}$. It is easy to see that $m_{\overline{P}} = \mathcal{F}^i(M_1, \dots, M_k, M')$ where we write $M' = M_{k+1} \upharpoonright (\mathcal{N}^{\text{com}} \cup \{A_i^{\text{cov}} : i \in \langle n \rangle\})$.

Then clearly $[p_{\epsilon,\ell_{k+1}}^c \mapsto [m_{\overline{P}}]] \in \mathcal{F}^\Pi(\epsilon, M' X_k M_k \cdots X_1 M_1)$ and we can deduce that $s^\Pi \ominus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m]] \oplus [p_{\epsilon,\ell_{k+1}}^c \mapsto [m_{\overline{P}}]] \in \mathcal{F}^\Pi(\epsilon M' X_k M_k \cdots X_1 M_1 \parallel \Pi_0)$.

Inspecting case **Rules (R'-6),(R'-7)** in the proof of R is a simulation we can see that $(m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\Gamma = \mathcal{F}^\Gamma(\Gamma(M_{k+1}))$ and $(m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\nu \in \mathcal{F}^\Pi(\Pi(M_{k+1}))$ where $\mathcal{G}^\Gamma = \{p_{c,msg}^s \mid c \in \text{Chan}, msg \in \text{Msg}\}$ and $\mathcal{G}^\nu = \{p_{\nu X}^s \mid X \in \mathcal{N}\}$. Further $(m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\Gamma \oplus (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\nu = (m_P \circ \zeta^{-1})$.

Hence we obtain $\mathcal{F}^\Gamma(\Gamma) \oplus (m_P \circ \zeta^{-1}) \upharpoonright \mathcal{G}^\Gamma = \mathcal{F}^\Gamma(\Gamma \oplus \Gamma(M_{k+1}))$. We can then conclude that $s_{j_1}^1 \in \mathcal{F}^\Pi(\epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \parallel \Pi(M_{k+1}) \triangleleft \Gamma \oplus \Gamma(M_{k+1}))$. From our assumptions above this implies (a) $m^1 = j^1$, (b) $m = 1$ and thus (c) $\Pi' \triangleleft \Gamma' = \epsilon M_{k+1} X_k M_k \cdots X_1 M_1 \parallel \Pi_0 \parallel \Pi(M_{k+1}) \triangleleft \Gamma \oplus \Gamma(M_{k+1})$. It is also easy to see that $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ and hence $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ which is what we wanted to prove. \square

Let us now turn to R^{-1} . Suppose $(\Pi \triangleleft \Gamma, \mathcal{F}(\Pi \triangleleft \Gamma)) \in R$ and $\mathcal{F}(\Pi \triangleleft \Gamma) \xrightarrow{\Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} S$. Hence $\mathcal{F}(\Pi \triangleleft \Gamma) \xrightarrow{\Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi' \triangleleft \Gamma') \xrightarrow{\epsilon}_{\mathcal{P}[\mathcal{N}]}^* S$. By definition this implies that there exists non-empty subsets of $Config$, $S(1), \dots, S(m)$ say, such that $m \geq 1$, $\mathcal{F}(\Pi \triangleleft \Gamma) = S(1) \xrightarrow{\epsilon}_{\mathcal{P}[\mathcal{N}]} S(2) \xrightarrow{\epsilon}_{\mathcal{P}[\mathcal{N}]} \dots \xrightarrow{\epsilon}_{\mathcal{P}[\mathcal{N}]} S(m-1) \xrightarrow{\Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} S(m) = \mathcal{F}(\Pi' \triangleleft \Gamma')$. Thus for $i \in \langle m \rangle$ we may pick configuration $s(i) \in S(i)$ such that for each $i \in \langle m-1 \rangle$ $s(i) = s_0 \xrightarrow{r^i(1)}_{\mathcal{N}} s_1 \xrightarrow{r^i(2)}_{\mathcal{N}} \dots \xrightarrow{r^i(m^{i-1})}_{\mathcal{N}} s_{m^{i-1}}^i \xrightarrow{r^i(m^i)}_{\mathcal{N}} s_{m^i}^i = s(i+1)$ where only one rule, $r^i(j^i)$ say, is not a weak spawn rule. We further know that for all $2 \leq i < m$ there *does not* exist $\Pi_0 \triangleleft \Gamma_0$ such that $s(i) \in \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$. Since clearly $s(1) \in \mathcal{F}(\Pi \triangleleft \Gamma)$ and $s(m) \in \mathcal{F}(\Pi' \triangleleft \Gamma')$ the Lemma above applies to give us $\Pi \triangleleft \Gamma \xrightarrow{\Pi' \triangleleft \Gamma'}^*_{\mathcal{G}} \Pi' \triangleleft \Gamma'$. We can thus conclude that R^{-1} is a weak simulation and hence R is a weak bisimulation.

Further suppose that we have $s \in \mathcal{F}(\Pi \triangleleft \Gamma)$ for some $\Pi \triangleleft \Gamma$ and $s' \in \mathcal{F}(\Pi' \triangleleft \Gamma')$ and $s \rightarrow_{\mathcal{N}}^* s'$ then we can decompose this path into a sequence of configurations $s(1) = s \rightarrow_{\mathcal{N}} s(2) \rightarrow_{\mathcal{N}} \dots \rightarrow_{\mathcal{N}} s(l) = s'$. Let $i_1, \dots, i_{l'}$ be the indices, in order, such that $s(i_j) \in \mathcal{F}(\Pi_j \triangleleft \Gamma_j)$ for some $\Pi_1 \triangleleft \Gamma_1, \dots, \Pi_{l'} \triangleleft \Gamma_{l'}$. Clearly for all $1 \leq j \leq l'$ we can apply the Lemma above to the path $s(i_j) \rightarrow_{\mathcal{N}}^* s(i_{j+1})$ to obtain $\Pi_1 \triangleleft \Gamma_1 \xrightarrow{\Pi_2 \triangleleft \Gamma_2}_{\mathcal{G}} \Pi_2 \triangleleft \Gamma_2 \xrightarrow{\Pi_3 \triangleleft \Gamma_3}_{\mathcal{G}} \dots \xrightarrow{\Pi_{l'} \triangleleft \Gamma_{l'}}_{\mathcal{G}} \Pi_{l'} \triangleleft \Gamma_{l'}$ which means that since R is a weak bisimulation that $\mathcal{F}(\Pi \triangleleft \Gamma) \xrightarrow{\Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi' \triangleleft \Gamma')$. Hence we can conclude that if $s \in \mathcal{F}(\Pi \triangleleft \Gamma)$ for some $\Pi \triangleleft \Gamma$ and $s' \in \mathcal{F}(\Pi' \triangleleft \Gamma')$ and $s \rightarrow_{\mathcal{N}}^* s'$ then $\mathcal{F}(\Pi \triangleleft \Gamma) \xrightarrow{\Pi' \triangleleft \Gamma'}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi' \triangleleft \Gamma')$.

We know that $\Pi_0 = A_1^0 \parallel \dots \parallel A_{n'}^0$, so let $s_{cov} = s_{Query} \oplus [p_{A_1}^{cov} \mapsto [\bullet], \dots, p_{A_n}^{cov} \mapsto [\bullet]] \oplus [p_{c, msg}^s \mapsto [\bullet^{c(c)(m)}] \mid c \in Chan, msg \in Msg]$ and $s_0 = s_{sim} \oplus [p_{A_i}^{cov} \mapsto [\bullet] \mid i \in \langle n' \rangle] \oplus [p_{c, msg}^s \mapsto [\bullet^{c(c)(m)}] \mid c \in Chan, msg \in Msg]$. We note that $s_0 \in \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$.

We will now show that $(\mathcal{N}, s_0, s_{cov})$ is a yes-instance of the coverability problem if and only if $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, A_1^{cov} \parallel \dots \parallel A_n^{cov} \triangleleft \Gamma^{cov})$ is a yes-instance of simple coverability.

Suppose $(\mathcal{N}, s_0, s_{cov})$ is a yes-instance of the coverability problem. Then there exists a $s' \in Config$ such that $s_0 \rightarrow_{\mathcal{N}}^* s'$ and $s_{cov} \leq_{Config} s'$. That means that there exists a configuration s'' such that $s_0 \rightarrow_{\mathcal{N}}^* s'', s'' \rightarrow_{\mathcal{N}} s(1) \rightarrow_{\mathcal{N}} \dots \rightarrow_{\mathcal{N}} s(l)$, $s(l) = s'$, $s'' \in \mathcal{F}(\Pi \triangleleft \Gamma)$ for some $\Pi \triangleleft \Gamma$, and for each $1 \leq i \leq l$ there does not exist a $\bar{\Pi} \triangleleft \bar{\Gamma}$ such that $s(i) \in \mathcal{F}(\bar{\Pi} \triangleleft \bar{\Gamma})$. We can thus appeal to our reasoning above to obtain that $\mathcal{F}(\Pi_0 \triangleleft \Gamma_0) \xrightarrow{\Pi \triangleleft \Gamma}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi \triangleleft \Gamma)$. We can decompose this path to get $\mathcal{F}(\Pi_0 \triangleleft \Gamma_0) \xrightarrow{\Pi_1 \triangleleft \Gamma_1}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi_1 \triangleleft \Gamma_1) \xrightarrow{\Pi_2 \triangleleft \Gamma_2}^*_{\mathcal{P}[\mathcal{N}]} \dots \xrightarrow{\Pi_{l'} \triangleleft \Gamma_{l'}}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi_{l'} \triangleleft \Gamma_{l'}) = \mathcal{F}(\Pi \triangleleft \Gamma)$. Since R is a weak bisimulation we thus know that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\Pi_1 \triangleleft \Gamma_1}_{\mathcal{G}} \Pi_1 \triangleleft \Gamma_1 \xrightarrow{\Pi_2 \triangleleft \Gamma_2}_{\mathcal{G}} \dots \xrightarrow{\Pi_{l'} \triangleleft \Gamma_{l'}}_{\mathcal{G}} \Pi_{l'} \triangleleft \Gamma_{l'} = \Pi \triangleleft \Gamma$ and thus $\Pi_0 \triangleleft \Gamma_0 \rightarrow_{\mathcal{G}}^* \Pi \triangleleft \Gamma$.

Let us turn to an inspection of s'' . We know that $s'' \in$

$\mathcal{F}(\Pi \triangleleft \Gamma)$, for all $1 \leq i \leq l$ there *does not* exist $\bar{\Pi} \triangleleft \bar{\Gamma}$ such that $s(i) \in \mathcal{F}(\bar{\Pi} \triangleleft \bar{\Gamma})$, and $s_{cov} \leq_{Config} s' = s(l)$. Since $s_{Query} \leq_{Config} s_{cov}$, we can conclude that $s_{Query} \leq_{Config} s'$. This implies that for some $0 \leq i < l$ we have the transition $s(j) \xrightarrow{r}_{\mathcal{N}} s(j+1)$ via the rule $r = (s_{sim}, s_{Query})$ which switches from simulation mode to query mode and where we write $s(0) = s''$. Inspecting the rules of \mathcal{N} we note that this mode switch cannot be reversed. We can deduce that $j = 0$ since using any other rule of \mathcal{N} for the transition $s'' \xrightarrow{r}_{\mathcal{N}} s(1)$ would either place $s(1) \in \mathcal{F}(\bar{\Pi} \triangleleft \bar{\Gamma})$ for some $\bar{\Pi} \triangleleft \bar{\Gamma}$ immediately, or start a CCFG computation disabling r until completion which would produce another $s(j) \in \mathcal{F}(\Pi_0 \triangleleft \Gamma_0)$ for some $\Pi_0 \triangleleft \Gamma_0$ before r could be enabled. Hence we can conclude that $s''(p_{A_i}^{cov}) = \emptyset$ for all $1 \leq i \leq n$ since $s'' \in \mathcal{F}(\Pi \triangleleft \Gamma)$ and for all $1 \leq j \leq l$ such that $s(j) \xrightarrow{r(j)}_{\mathcal{N}} s(j+1)$ the rule $r(j)$ is a rule of the form $((p_{A_i}^{cov}, \ell_{k+1} \Xi, s_{Query}), (p_{\epsilon, (-)} \Xi, \emptyset, s_{Query} \oplus [p_{A_i}^{cov} \mapsto [\bullet]]))$ or $((p_{\epsilon, A_i}^{cov} \Xi, s_{Query}), (p_{\epsilon, (-)} \Xi, \emptyset, s_{Query} \oplus [p_{A_i}^{cov} \mapsto [\bullet]]))$. Since $s_{cov} \leq_{Config} s'$ this means that at least there are indices i_1, \dots, i_n (not necessarily in order) where for each $1 \leq j \leq n$ rule $r(i_j)$ places a \bullet -token into place $p_{A_j}^{cov}$. Since the rules $r(1), \dots, r(l)$ only remove complex tokens from places of the form $p_{A, \ell_{k+1} \Xi}^C$ or $p_{\epsilon, A \Xi}^C$ we can conclude that there exists complex tokens m_1, \dots, m_n located in places p_1, \dots, p_n in configuration s'' such that for all $1 \leq j \leq n$ the place $p_j = p_{A_j, \ell_{k+1} \Xi}^C$ or $p_{\epsilon, A_j \Xi}^C$. Since $s'' \in \mathcal{F}(\Pi \triangleleft \Gamma)$ this implies there exists processes π_1, \dots, π_n such that $\Pi = \pi_1 \parallel \dots \parallel \pi_n \parallel \Pi_0$ and each $\pi_j = A_j^{cov} \cdot \gamma_j$ or $\epsilon M_j \cdot \gamma_j$ with $M_j(A_j^{cov}) > 0$ for $1 \leq j \leq n$. Further it is also easy to see that Γ^{cov} is covered by Γ . Hence we can deduce that $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, A_1^{cov} \parallel \dots \parallel A_n^{cov} \triangleleft \Gamma^{cov})$ is a yes-instance of simple coverability for the alternative semantics.

For the other direction of the reduction suppose $(\mathcal{G}, \Pi_0 \triangleleft \Gamma_0, A_1^{cov} \parallel \dots \parallel A_n^{cov} \triangleleft \Gamma^{cov})$ is a yes-instance of simple coverability in the alternative semantics. This means that $\Pi_0 \triangleleft \Gamma_0 \xrightarrow{\Pi_1 \triangleleft \Gamma_1}_{\mathcal{G}} \Pi_1 \triangleleft \Gamma_1 \xrightarrow{\Pi_2 \triangleleft \Gamma_2}_{\mathcal{G}} \dots \xrightarrow{\Pi_l \triangleleft \Gamma_l}_{\mathcal{G}} \Pi_l \triangleleft \Gamma_l =: \Pi \triangleleft \Gamma$ where $\Pi = \pi_1 \parallel \dots \parallel \pi_n \parallel \Pi_0$ and each $\pi_j = A_j^{cov} \cdot \gamma_j$ or $\epsilon M_j \cdot \gamma_j$ with $M_j(A_j^{cov}) > 0$ for $1 \leq j \leq n$.

Since R is a weak bisimulation we obtain $\mathcal{F}(\Pi_0 \triangleleft \Gamma_0) \xrightarrow{\Pi_1 \triangleleft \Gamma_1}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi_1 \triangleleft \Gamma_1) \xrightarrow{\Pi_2 \triangleleft \Gamma_2}^*_{\mathcal{P}[\mathcal{N}]} \dots \xrightarrow{\Pi_{l-1} \triangleleft \Gamma_{l-1}}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi_{l-1} \triangleleft \Gamma_{l-1}) \xrightarrow{\Pi \triangleleft \Gamma}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi \triangleleft \Gamma)$.

Thus we may pick $s(1), \dots, s(l)$ such that for all $1 \leq j \leq l$ we have $s(j) \in \mathcal{F}(\Pi_j \triangleleft \Gamma_j)$ and for $j < l$ we have $s(j) \rightarrow_{\mathcal{N}} s(j+1)$. Hence $s(1) \rightarrow_{\mathcal{N}} s(l)$. Further since $\mathcal{F}(\Pi_0 \triangleleft \Gamma_0) \xrightarrow{\Pi_1 \triangleleft \Gamma_1}^*_{\mathcal{P}[\mathcal{N}]} \mathcal{F}(\Pi_1 \triangleleft \Gamma_1)$ and $s_0 \rightarrow_{\mathcal{N}}^* s$ for all $s \in \mathcal{F}(S \triangleleft \emptyset)$ we can conclude that $s_0 \rightarrow_{\mathcal{N}}^* s(l) =: s'$.

Let us now inspect s' . Since $s' \in \mathcal{F}(\Pi \triangleleft \Gamma)$ we can deduce that there exists complex tokens m_1, \dots, m_n located in places p_1, \dots, p_n in configuration s' such that for all $1 \leq j \leq n$ the place $p_j = p_{A_j, \ell_{k+1} \Xi}^C$ or $p_{\epsilon, A_j \Xi}^C$. Hence we may extend the derivation $s_0 \rightarrow_{\mathcal{N}}^* s' \xrightarrow{r}_{\mathcal{N}} s'_0 \xrightarrow{r(1)}_{\mathcal{N}} s'_1 \xrightarrow{r(2)}_{\mathcal{N}} \dots \xrightarrow{r(n)}_{\mathcal{N}} s'_n$ where $r = (s_{sim}, s_{Query})$ switches from simulation mode to query mode and the rule $r(j) =$

$((p_{A_j}^{C_{\text{cov}}, \ell_{k+1}} \Xi, s_{\text{Query}}), (p_{\epsilon, (-)}^C \Xi, \emptyset, s_{\text{Query}} \oplus [p_{A_j}^{C_{\text{cov}}} \mapsto [\bullet]]))$ or $((p_{\epsilon, A_j}^{C_{\text{cov}}} \Xi, s_{\text{Query}}), (p_{\epsilon, (-)}^C \Xi, \emptyset, s_{\text{Query}} \oplus [p_{A_j}^{C_{\text{cov}}} \mapsto [\bullet]]))$ and thus we can deduce that s'_n has one \bullet -token located in each place $p_{A_1}^{C_{\text{cov}}}, \dots, p_{A_n}^{C_{\text{cov}}}$ and s_{Query} as a submarking, i.e. $s_{\text{cov}} \leq_{\text{Config}} s'_n$. Hence we can deduce that $(\mathcal{N}, s_0, s_{\text{cov}})$ is a yes-instance of the coverability problem.

Hence we can conclude that the simple coverability problem for K -shaped APCPS in the alternative semantics EXPTIME reduces to the coverability problem of NNCT. \square

Theorem 4. *Simple coverability, boundedness and termination for a total-transfer NNCT EXPTIME reduces to simple coverability, boundedness and termination respectively for a 4-shaped APCPS in the alternative semantics.*

Proof. Fix a total-transfer NNCT $\mathcal{N} = (\mathcal{P}^S, \mathcal{P}^C, \mathcal{P}^{\text{col}}, \mathcal{R}, \zeta)$ and an instance of coverability $(\mathcal{N}, s_0, s_{\text{cov}})$ with a simple query, i.e. $s_0(p) = s_{\text{cov}}(p) = \emptyset$ for all $p \in \mathcal{P}^C$. Let us also fix an enumeration of $\mathcal{P}^S = \{p_{(1)}, \dots, p_{(n_s)}\}$, $\mathcal{P}^C = \{p'_{(1)}, \dots, p'_{(n_c)}\}$ and $\mathcal{P}^{\text{col}} = \{p^{\text{col}}_{(1)}, \dots, p^{\text{col}}_{(n_{\text{col}})}\}$. Since \mathcal{N} is a total-transfer NNCT we know that for all $r \in \text{TransferRules}$ it is the case that $r = ((p, I), (p', \mathcal{P}^{\text{col}}, O))$.

Let us define an APCPS $\mathcal{G} = (\Sigma, I, \mathcal{N}, \mathcal{R}, S)$ that will simulate \mathcal{N} . The APCPS \mathcal{G} has a channel c_p for each simple or complex place p plus a special channel $c_?$ and let the set of channels be $\text{Chan} = \{c_p \mid p \in \mathcal{P}^S \cup \mathcal{P}^C\} \cup \{c_?\}$. For \mathcal{G} 's messages let us first inspect \mathcal{N} 's complex rules: let Ξ be the set of complex tokens representing “injected” coloured tokens in \mathcal{N} 's complex rules plus the set of complex tokens that are created by simple rules, i.e. $\Xi = \{c \mid ((p, I), (p', c, O)) \in \text{ComplexRules}\} \cup \{c \mid p \in \mathcal{P}^C, (I, O) \in \text{SimpleRules}, O(p)(c) \geq 1\}$ then \mathcal{G} will have a message $m_{p,d}$ for each $p \in \mathcal{P}^C$ and $d \in \Xi$ and two special messages $m_{p,\emptyset}, m_{p,\downarrow}$ for each $p \in \mathcal{P}^C$. Further \mathcal{G} will have a message \bullet ; and hence we can let the set of messages be $\text{Msg} = \{msg_{p,d} \mid p \in \mathcal{P}^C, d \in \Xi \cup \{\emptyset, \downarrow\}\} \cup \{\bullet\}$.

The APCPS \mathcal{G} will have non-terminals N_r, N_r^I for each rule $r \in \mathcal{R}$, non-terminals $N_r^{s,o}, N_r^{c,o}$ for each $r \in \text{SimpleRules}$, and non-terminals N_r^o, N_r^c for each $r \in \text{ComplexRules} \cup \text{TransferRules}$. Additionally \mathcal{G} will have non-terminals $N^{\nu p}, N^p$ for every $p \in \mathcal{P}^C$, a non-terminal N^c for each $c \in \Xi$ and two special non-terminals $N^{\nu?}$ and N^{sim} . Hence \mathcal{G} 's set of non-terminals are

$$\begin{aligned} \mathcal{N} = & \{N^{\nu p}, N^p \mid p \in \mathcal{P}^C\} \cup \{N^c \mid c \in \Xi\} \cup \{N^{\nu?}, N^{\text{sim}}\} \\ & \cup \{N_r, N_r^I \mid r \in \mathcal{R}\} \cup \{N_r^{s,o}, N_r^{c,o} \mid r \in \text{SimpleRules}\} \\ & \cup \{N_r^o, N_r^c \mid r \in \text{ComplexRules} \cup \text{TransferRules}\} \end{aligned}$$

Let us also define a designated non-terminal A_{cov} to label the coverability query. We can then set \mathcal{G} 's alphabet $\Sigma = \{c!msg, c?msg, \nu X \mid c \in \text{Chan}, msg \in \text{Msg}, X \in \mathcal{N}\}$. Further let us use the standard independence relation for APCPS I over $\Sigma \cup \mathcal{N}$.

A configuration $s \in \text{Config}$ will be represented by an APCPS configuration in the following way:

- for each simple place $p \in \mathcal{P}^S$, the channel c_p will contain $|s(p)|$ \bullet -messages — we can formalise

this by a function \mathcal{F}^Γ that we define as $\mathcal{F}^\Gamma(s) = [\bullet^{|s(p_{(1)})|}]^{c_{p_{(1)}}}, \dots, [\bullet^{|s(p_{(n_s)})|}]^{c_{p_{(n_s)}}}$;

- for each complex place $p \in \mathcal{P}^C$, suppose $s(p) = [m_1, \dots, m_k]$ then for each m_i there will be one process with a process-state $N^p \cdot \tilde{\Gamma}(m_i \circ \zeta^{-1}) \cdot N^{\nu?}$ where $\tilde{\Gamma}$ is a function that takes a mapping $m_0 : \mathcal{P}^S \rightarrow \mathbb{N}$ and transforms it into a multiset $\mathbb{M}[c_{\mathcal{P}^S}! \bullet]$ by $\tilde{\Gamma}(m_0)(c_p! \bullet) = |m_0(p)|$ — we can formalise this with two functions $\mathcal{F}^\Pi(s, p) = \prod_{i=1}^k N^p \cdot \tilde{\Gamma}(m_i \circ \zeta^{-1}) \cdot N^{\nu?}$ and $\mathcal{F}^\Pi(s) = \prod_{i=1}^{n_c} \mathcal{F}^\Pi(s, p'_{(i)})$;
- in addition we have one administrative process, that implements the execution of \mathcal{N} 's rules and is in the process-state N^{sim} when representing the configuration s .

Formally, we define a representation function \mathcal{F} by $\mathcal{F}(s) = N^{\text{sim}} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s)$ and define the relation $R \subseteq \text{Config} \times \text{Config}^{\text{APCPS}}$ by $R = \{(s, \mathcal{F}(s)) \mid s \in \text{Config}\}$. We will prove that R is a weak bisimulation for $(\text{Config}, \rightarrow_{\mathcal{N}})$ and $(\text{Config}^{\text{APCPS}}, \rightarrow_{\mathcal{G}})$ where we write $\rightarrow_{\mathcal{G}}$ for \rightarrow_{con} here and in the following.

We will now define and explain how the administrative process is implementing \mathcal{N} 's rules.

For each rule $r \in \mathcal{R}$ the APCPS \mathcal{G} has the rule $N^{\text{sim}} \rightarrow N_r, N^{\text{sim}}$ which guesses one of \mathcal{N} 's rules to execute next. Depending on whether r is a simple, complex or transfer rule the implementation is different:

- $r = (I, O) \in \text{SimpleRules}$

If r is a simple rule, then N_r rewrites to three non-terminals: N_r^I which removes \bullet -messages as described by I , $N_r^{o,s}$ which sends \bullet -messages as described by the effect of O on \mathcal{N} 's simple places and $N_r^{o,c}$ which spawns new processes which will represent the newly created complex tokens as described by O on \mathcal{N} 's complex places. Let us enumerate the set $\Xi = \{c_{(1)}, \dots, c_{(n_\Xi)}\}$ then we can formally implement the above by the following rules:

$$\begin{aligned} N_r & \rightarrow N_r^I \cdot N_r^{s,o} \cdot N_r^{c,o} \\ N_r^I & \rightarrow (c_{p_{(1)}} ? \bullet)^{|I(p_{(1)})|} \dots (c_{p_{(n_s)}} ? \bullet)^{|I(p_{(n_s)})|} \\ N_r^{s,o} & \rightarrow (c_{p_{(1)}} ! \bullet)^{|O(p_{(1)})|} \dots (c_{p_{(n_s)}} ! \bullet)^{|O(p_{(n_s)})|} \\ N_r^{c,o} & \rightarrow (\nu N^{\nu p_{(1)}} (c_? ! msg_{p'_{(1)}, c_{(1)}}) (c_? ? \bullet))^{O(p_{(1)})(c_{(1)})} \dots \\ & \quad (\nu N^{\nu p'_{(i)}} (c_? ! msg_{p'_{(i)}, c_{(i)}}) (c_? ? \bullet))^{O(p'_{(i)})(c_{(i)})} \dots \\ & \quad (\nu N^{\nu p_{(n_c)}} (c_? ! msg_{p'_{(n_c)}, c_{(n_\Xi)}}) (c_? ? \bullet))^{O(p'_{(n_c)})(c_{(n_\Xi)})} \end{aligned}$$

and the rules for complex token representing processes we have the following rule for each $c \in \Xi$ and $p \in \mathcal{P}^C$:

$$N^{\nu p} \rightarrow (c_? ? msg_{p,c}) (c_? ! \bullet) N^p \cdot N^c \cdot N^{\nu?}$$

and for each $c \in \Xi$:

$$N^c \rightarrow (c_{\zeta(p_{(1)}^{\text{col}})} ! \bullet)^{|c(p_{(1)}^{\text{col}})|} \dots (c_{\zeta(p_{(n_{\text{col}})}^{\text{col}})} ! \bullet)^{|c(p_{(n_{\text{col}})}^{\text{col}})|}$$

- $r = ((p, I), (p', c, O)) \in \text{ComplexRules}$

If r is a complex rule, then N_r rewrites to three non-terminals: N_r^I which removes \bullet -messages as described by

I , N_r^O which sends \bullet -messages as described by O and N_r^C which forces one process representing one complex token m in complex place p to change state so that the process afterwards represents $m \oplus c$ in complex place p' . Formally we can implement this with the following rules:

$$\begin{aligned} N_r &\rightarrow N_r^I \cdot N_r^O \cdot N_r^C \\ N_r^I &\rightarrow (c_{p(1)} ? \bullet)^{|I(p(1))|} \dots (c_{p(n_S)} ? \bullet)^{|I(p(n_S))|} \\ N_r^O &\rightarrow (c_{p(1)} ! \bullet)^{|O(p(1))|} \dots (c_{p(n_S)} ! \bullet)^{|O(p(n_S))|} \\ N_r^C &\rightarrow (c_p ! msg_{p',c}) \cdot (c_p ? \bullet) \end{aligned}$$

and for each $c \in \Xi$ and $p \in \mathcal{P}^C$ we have the following rule for complex token representing processes:

$$N^p \rightarrow (c_p ? msg_{p',c}) \cdot (c_p ! \bullet) \cdot N^{p'} N^c$$

- $r = ((p, I), (p', \mathcal{P}^{col}, O)) \in TransferRules$

If r is a transfer rule, then N_r rewrites to three non-terminals: N_r^I which removes \bullet -messages as described by I , N_r^O which sends \bullet -messages as described by O and N_r^C which forces one process representing one complex token m in complex place p to change state and make its summary effective, which transfers $(m \circ \zeta^{-1})$ to the channels, so that the process afterwards represents the empty complex token \emptyset in complex place p' . Formally we can implement this with the following rules:

$$\begin{aligned} N_r &\rightarrow N_r^I \cdot N_r^O \cdot N_r^C \\ N_r^I &\rightarrow (c_{p(1)} ? \bullet)^{|I(p(1))|} \dots (c_{p(n_S)} ? \bullet)^{|I(p(n_S))|} \\ N_r^O &\rightarrow (c_{p(1)} ! \bullet)^{|O(p(1))|} \dots (c_{p(n_S)} ! \bullet)^{|O(p(n_S))|} \\ N_r^C &\rightarrow (c_p ! msg_{p,\downarrow}) \cdot (c_p ? \bullet) \cdot (c_p ? ! msg_{p',\emptyset}) \cdot (c_p ? \bullet) \end{aligned}$$

and for each $p \in \mathcal{P}^C$ we have the following rule for complex token representing processes:

$$\begin{aligned} N^p &\rightarrow (c_p ? msg_{p,\downarrow}) \cdot (c_p ! \bullet) \\ N^{p?} &\rightarrow (c_p ? msg_{p,\emptyset}) \cdot (c_p ? ! \bullet) \cdot N^p \cdot N^{p?} \end{aligned}$$

Further we add two more rules:

$$\begin{aligned} S &\rightarrow (c_{p(1)} ! \bullet)^{|s_0(p(1))|} \dots (c_{p(n_S)} ! \bullet)^{|s_0(p(n_S))|} N^{sim} \\ N^{sim} &\rightarrow (c_{p(1)} ? \bullet)^{|s_{cov}(p(1))|} \dots (c_{p(n_S)} ? \bullet)^{|s_{cov}(p(n_S))|} A_{cov}. \end{aligned}$$

The first rule simply sets up the simulation from s_0 . The second rule has the purpose to encode \mathcal{N} 's coverability query with the intention that a configuration $A_{cov} \parallel \Pi \triangleleft \Gamma$ is only reachable if and only if s_{cov} is coverable.

First, let us note that clearly \mathcal{G} can be constructed from \mathcal{N} in EXPTIME. Also all non-terminals except for the N_r^O , $N_r^{O,S}$ and N^c are non-commutative. It is easy to see that \mathcal{G} has shaped stacks since the only non-terminal loop increasing the “call-stack” is in the rule

$$N^p \rightarrow (c_p ? msg_{p',c}) \cdot (c_p ! \bullet) \cdot N^{p'} N^c$$

and N^c is a commutative non-terminal. It is easy to see that picking $K = 4$ is adequate.

Before we go on to prove R is a weak bisimulation let us first analyse \mathcal{F}^Π and \mathcal{F}^Γ .

It is easy to see that for all $s, s' \in Config$ we have

$$\begin{aligned} \mathcal{F}^\Pi(s) \parallel \mathcal{F}^\Pi(s') &= \mathcal{F}^\Pi(s \oplus s'), \\ \mathcal{F}^\Gamma(s) \oplus \mathcal{F}^\Gamma(s') &= \mathcal{F}^\Gamma(s \oplus s'), \end{aligned}$$

if for all $p \in \mathcal{P}^S$ we have $s'(p) = \emptyset$ then

$$\mathcal{F}^\Gamma(s) = \mathcal{F}^\Gamma(s \oplus s') = \mathcal{F}^\Gamma(s \oplus s')$$

and if for all $p \in \mathcal{P}^C$ we have $s'(p) = \emptyset$ then

$$\mathcal{F}^\Pi(s) = \mathcal{F}^\Pi(s \oplus s') = \mathcal{F}^\Pi(s \oplus s')$$

Let us label APCPS and NNCT transitions in the following way: for $s, s' \in Config$ such that $s \rightarrow_{\mathcal{N}} s'$ let us label $\rightarrow_{\mathcal{N}}$ such that we write $s \xrightarrow{s'}_{\mathcal{N}} s'$; for $\Pi \triangleleft \Gamma, \Pi' \triangleleft \Gamma' \in Config^{APCPS}$ such that $\Pi \triangleleft \Gamma \rightarrow_{\mathcal{G}} \Pi' \triangleleft \Gamma'$ and $\exists s_0 \in Config$ such that $\mathcal{F}(s_0) = \Pi' \triangleleft \Gamma'$ let us label $\rightarrow_{\mathcal{G}}$ such that we write $\Pi \triangleleft \Gamma \xrightarrow{s_0}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$; if however $\nexists s_0 \in Config$ such that $\mathcal{F}(s_0) = s'$ then let us label $\rightarrow_{\mathcal{G}}$ such that we write $\Pi \triangleleft \Gamma \xrightarrow{\epsilon}_{\mathcal{G}} \Pi' \triangleleft \Gamma'$.

Let us clarify the definition of a weak bisimulation for a labelled transition system.

Definition ((Weak) simulation, bisimulation). Suppose (S, \rightarrow_S) and $(S', \rightarrow_{S'})$ are labelled transition systems we say a relation $\mathfrak{R} \subseteq S \times S'$ is a (weak) *simulation* just if for all $(s, s') \in \mathfrak{R}$, if for some $t \in S$ we have $s \xrightarrow{\alpha}_S t$ ($s \xrightarrow{\alpha}_S^* t$) then there exists $t' \in S'$ such that $s' \xrightarrow{\alpha}_{S'} t'$ ($s' \xrightarrow{\alpha}_{S'}^* t'$) and $(t, t') \in \mathfrak{R}$. We say \mathfrak{R} is a (weak) *bisimulation* just if both \mathfrak{R} and \mathfrak{R}^{-1} are (weak) simulations.

Let us clarify some notation: if $\Gamma \in (Chan \rightarrow \mathbb{M}[Msg])$ such that for $\Gamma', \Gamma'' \in (Chan \rightarrow \mathbb{M}[Msg])$ we can decompose Γ such that $\Gamma = \Gamma' \oplus \Gamma''$ then $\Gamma' = \Gamma \ominus \Gamma''$.

Let us now prove that R above is a weak simulation. So suppose we have $(s, \mathcal{F}(s)) \in R$ and $s \xrightarrow{s_0}_{\mathcal{N}} s'$ then from the labelling we clearly have $s \xrightarrow{s'}_{\mathcal{N}} s'$. Hence there exists a $r \in \mathcal{R}$ such that $s \xrightarrow{r}_{\mathcal{N}} s'$. We will perform a case analysis on r .

- *Case:* $r \in SimpleRules$.

Then $r = (I, O)$ and $s' = (s \oplus I) \oplus O$ and also $s \oplus I \in Config$. We know that $\mathcal{F}(s) = N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s)$ and since $s \oplus I \in Config$ we have that $|s(p_0)| \geq |I(p_0)|$ for all $p_0 \in \mathcal{P}^S$. Hence by definition

$$\mathcal{F}^\Gamma(s) = \Gamma \oplus \left[\bullet^{|I(p(1))|} \right]^{c_{p(1)}} \oplus \dots \oplus \left[\bullet^{|I(p(n_S))|} \right]^{c_{p(n_S)}}$$

for some $\Gamma \in (Chan \rightarrow \mathbb{M}[Msg])$. Hence we can see that we can perform the following transitions:

$$\begin{aligned} \mathcal{F}(s) &\xrightarrow{\epsilon}_{\mathcal{G}} N_r N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s) \\ &\xrightarrow{\epsilon}_{\mathcal{G}} N_r^I N_r^{O,S} N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s) \\ &\xrightarrow{\epsilon}_{\mathcal{G}} (c_{p(1)} ? \bullet)^{|I(p(1))|} \dots (c_{p(n_S)} ? \bullet)^{|I(p(n_S))|} \\ &\quad N_r^{O,S} N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s) \\ &\xrightarrow{\epsilon}_{\mathcal{G}} (c_{p(1)} ? \bullet)^{|I(p(1))|-1} \dots (c_{p(n_S)} ? \bullet)^{|I(p(n_S))|} \\ &\quad N_r^{O,S} N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s) \ominus [\bullet]^{c_{p(1)}} \\ &\xrightarrow{\epsilon}_{\mathcal{G}} \dots \end{aligned}$$

$$\begin{aligned}
& \xrightarrow{\epsilon}_{\mathcal{G}} (c_{p(2)} ? \bullet)^{|I(p(2))|} \dots (c_{p(n_S)} ? \bullet)^{|I(p(n_S))|} \\
& N_r^{O,S} N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s) \ominus \left[\bullet^{I(p(1))} \right]^{c_{p(1)}} \\
& \xrightarrow{\epsilon}_{\mathcal{G}} \dots \\
& \xrightarrow{\epsilon}_{\mathcal{G}} N_r^{O,S} N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s) \ominus \left[\bullet^{I(p(1))} \right]^{c_{p(1)}} \\
& \quad \ominus \left[\bullet^{I(p(2))} \right]^{c_{p(2)}} \ominus \dots \ominus \left[\bullet^{I(p(n_S))} \right]^{c_{p(n_S)}} \\
& = N_r^{O,S} N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I)
\end{aligned}$$

We can expand $N_r^{O,S}$ and perform the send actions similarly to the way we expanded N_r^I and its receive actions.

$$\begin{aligned}
\mathcal{F}(s) & \xrightarrow{\epsilon}_{\mathcal{G}}^* (c_{p(1)} ! \bullet)^{|O(p(1))|} \dots (c_{p(n_S)} ! \bullet)^{|O(p(n_S))|} \\
& N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I) \\
& \xrightarrow{\epsilon}_{\mathcal{G}}^* N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I) \oplus \left[\bullet^{O(p(1))} \right]^{c_{p(1)}} \\
& \quad \oplus \dots \oplus \left[\bullet^{O(p(n_S))} \right]^{c_{p(n_S)}} \\
& = N_r^{O,C} N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S))
\end{aligned}$$

We continue with expanding $N_r^{O,C}$. Similarly to above we can then perform all the spawns and synchronisations.

$$\begin{aligned}
\mathcal{F}(s) & \xrightarrow{\epsilon}_{\mathcal{G}} \left(\nu N^{\nu p'(1)} \left(c ? ! msg_{p'(1), \epsilon(1)} \right) (c ? ? \bullet) \right)^{O(p'(1))(\epsilon(1))} \dots \\
& \quad \left(\nu N^{\nu p'(i)} \left(c ? ! msg_{p'(i), \epsilon(i)} \right) (c ? ? \bullet) \right)^{O(p'(i))(\epsilon(i))} \dots \\
& \quad \left(\nu N^{\nu p'(n_C)} \left(c ? ! msg_{p'(n_C), \epsilon(n_\Xi)} \right) (c ? ? \bullet) \right)^{O(p'(n_C))(\epsilon(n_\Xi))} \\
& N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S)) \\
& \xrightarrow{\epsilon}_{\mathcal{G}} \left(c ? ! msg_{p'(1), \epsilon(1)} \right) (c ? ? \bullet) \\
& \quad \left(\nu N^{\nu p'(1)} \left(c ? ! msg_{p'(1), \epsilon(1)} \right) (c ? ? \bullet) \right)^{O(p'(1))(\epsilon(1)) - 1} \dots \\
& \quad \left(\nu N^{\nu p'(i)} \left(c ? ! msg_{p'(i), \epsilon(i)} \right) (c ? ? \bullet) \right)^{O(p'(i))(\epsilon(i))} \dots \\
& \quad \left(\nu N^{\nu p'(n_C)} \left(c ? ! msg_{p'(n_C), \epsilon(n_\Xi)} \right) (c ? ? \bullet) \right)^{O(p'(n_C))(\epsilon(n_\Xi))} \\
& N^{sim} \parallel N^{\nu p'(1)} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S))
\end{aligned}$$

Using the rule $N^{\nu p'(1)} \rightarrow (c ? ? msg_{p, \epsilon(1)}) (c ? ! \bullet) N^{p'(1)} \cdot N^{\epsilon(1)} \cdot N^{\nu ?}$ we can derive

$$\begin{aligned}
N^{\epsilon(1)} & \rightarrow^* \left(c_{\zeta(p_{(1)}^{col})} ! \bullet \right)^{|c(1)(p_{(1)}^{col})|} \dots \left(c_{\zeta(p_{(n_S)}^{col})} ! \bullet \right)^{|c(1)(p_{(n_S)}^{col})|} \\
& =: w
\end{aligned}$$

and the equality $\mathbb{M}(w) = \tilde{\Gamma}(\epsilon(1) \circ \zeta^{-1})$ holds. Thus

$$\begin{aligned}
\mathcal{F}(s) & \xrightarrow{\epsilon}_{\mathcal{G}}^* \left(c ? ! msg_{p'(1), \epsilon(1)} \right) (c ? ? \bullet) \\
& \quad \left(\nu N^{\nu p'(1)} \left(c ? ! msg_{p'(1), \epsilon(1)} \right) (c ? ? \bullet) \right)^{O(p'(1))(\epsilon(1)) - 1} \dots \\
& \quad \left(\nu N^{\nu p'(i)} \left(c ? ! msg_{p'(i), \epsilon(i)} \right) (c ? ? \bullet) \right)^{O(p'(i))(\epsilon(i))} \dots \\
& \quad \left(\nu N^{\nu p'(n_C)} \left(c ? ! msg_{p'(n_C), \epsilon(n_\Xi)} \right) (c ? ? \bullet) \right)^{O(p'(n_C))(\epsilon(n_\Xi))}
\end{aligned}$$

$$\begin{aligned}
& N^{sim} \parallel \left(c ? ? msg_{p, \epsilon} \right) (c ? ! \bullet) N^{p'(1)} \cdot \tilde{\Gamma}(\epsilon(1) \circ \zeta^{-1}) \cdot N^{\nu ?} \\
& \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S)) \\
& \xrightarrow{\epsilon}_{\mathcal{G}}^* \left(\nu N^{\nu p'(1)} \left(c ? ! msg_{p'(1), \epsilon(1)} \right) (c ? ? \bullet) \right)^{O(p'(1))(\epsilon(1)) - 1} \dots \\
& \quad \left(\nu N^{\nu p'(i)} \left(c ? ! msg_{p'(i), \epsilon(i)} \right) (c ? ? \bullet) \right)^{O(p'(i))(\epsilon(i))} \dots \\
& \quad \left(\nu N^{\nu p'(n_C)} \left(c ? ! msg_{p'(n_C), \epsilon(n_\Xi)} \right) (c ? ? \bullet) \right)^{O(p'(n_C))(\epsilon(n_\Xi))} \\
& N^{sim} \parallel N^{p'(1)} \cdot \tilde{\Gamma}(\epsilon(1) \circ \zeta^{-1}) \cdot N^{\nu ?} \\
& \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S))
\end{aligned}$$

we can continue like this:

$$\begin{aligned}
\mathcal{F}(s) & \xrightarrow{\epsilon}_{\mathcal{G}}^* \left(\nu N^{\nu p'(1)} \left(c ? ! msg_{p'(1), \epsilon(2)} \right) (c ? ? \bullet) \right)^{O(p'(1))(\epsilon(2))} \dots \\
& \quad \left(\nu N^{\nu p'(i)} \left(c ? ! msg_{p'(i), \epsilon(i)} \right) (c ? ? \bullet) \right)^{O(p'(i))(\epsilon(i))} \dots \\
& \quad \left(\nu N^{\nu p'(n_C)} \left(c ? ! msg_{p'(n_C), \epsilon(n_\Xi)} \right) (c ? ? \bullet) \right)^{O(p'(n_C))(\epsilon(n_\Xi))} \\
& N^{sim} \parallel \prod_{i=1}^{O(p'(1))(\epsilon(1))} N^{p'(1)} \cdot \tilde{\Gamma}(\epsilon(1) \circ \zeta^{-1}) \cdot N^{\nu ?} \\
& \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S)) \\
& \xrightarrow{\epsilon}_{\mathcal{G}}^* \left(\nu N^{\nu p'(2)} \left(c ? ! msg_{p'(2), \epsilon(1)} \right) (c ? ? \bullet) \right)^{O(p'(2))(\epsilon(1))} \dots \\
& \quad \left(\nu N^{\nu p'(i)} \left(c ? ! msg_{p'(i), \epsilon(i)} \right) (c ? ? \bullet) \right)^{O(p'(i))(\epsilon(i))} \dots \\
& \quad \left(\nu N^{\nu p'(n_C)} \left(c ? ! msg_{p'(n_C), \epsilon(n_\Xi)} \right) (c ? ? \bullet) \right)^{O(p'(n_C))(\epsilon(n_\Xi))} \\
& N^{sim} \parallel \prod_{i=1}^{O(p'(1))(\epsilon(1))} N^{p'(1)} \cdot \tilde{\Gamma}(\epsilon(1) \circ \zeta^{-1}) \cdot N^{\nu ?} \\
& \parallel \dots \\
& \parallel \prod_{i=1}^{O(p'(1))(\epsilon(n_\Xi))} N^{p'(1)} \cdot \tilde{\Gamma}(\epsilon(n_\Xi) \circ \zeta^{-1}) \cdot N^{\nu ?} \\
& \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S)) \\
& = \left(\nu N^{\nu p'(2)} \left(c ? ! msg_{p'(2), \epsilon(1)} \right) (c ? ? \bullet) \right)^{O(p'(2))(\epsilon(1))} \dots \\
& \quad \left(\nu N^{\nu p'(i)} \left(c ? ! msg_{p'(i), \epsilon(i)} \right) (c ? ? \bullet) \right)^{O(p'(i))(\epsilon(i))} \dots \\
& \quad \left(\nu N^{\nu p'(n_C)} \left(c ? ! msg_{p'(n_C), \epsilon(n_\Xi)} \right) (c ? ? \bullet) \right)^{O(p'(n_C))(\epsilon(n_\Xi))} \\
& N^{sim} \parallel \mathcal{F}^\Pi(O, p'(1)) \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S)) \\
& \xrightarrow{\alpha}_{\mathcal{G}}^* N^{sim} \parallel \mathcal{F}^\Pi(s) \parallel \left(\prod_{i=1}^{n_C} \mathcal{F}^\Pi(O, p'(i)) \right) \\
& \quad \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S)) \\
& = N^{sim} \parallel \mathcal{F}^\Pi(s) \parallel \mathcal{F}^\Pi(O \upharpoonright \mathcal{P}^C) \\
& \quad \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^S))
\end{aligned}$$

It just remains to analyse the last configuration:

$$\begin{aligned}
\mathcal{F}^\Pi(s) \parallel \mathcal{F}^\Pi(O \upharpoonright \mathcal{P}^C) & = \mathcal{F}^\Pi(s) \parallel \mathcal{F}^\Pi(O) = \mathcal{F}^\Pi(s \oplus O) \\
& = \mathcal{F}^\Pi(s \ominus I \oplus O) = \mathcal{F}^\Pi(s')
\end{aligned}$$

and

$$\mathcal{F}^\Gamma(s \ominus I \oplus (O \upharpoonright \mathcal{P}^s)) = \mathcal{F}^\Gamma(s \ominus I \oplus O) = \mathcal{F}^\Gamma(s')$$

from which we can deduce $\mathcal{F}(s) \xrightarrow{\alpha}_G^* \mathcal{F}(s')$, $(s', \mathcal{F}(s')) \in R$ and clearly $\alpha = s'$ which is what we wanted to prove.

- *Case: $r \in \text{ComplexRules}$.*

Then $r = ((p, I), (p', \oplus, O))$ and $s' = (s \ominus I \oplus [p \mapsto [m]]) \oplus O \oplus [p' \mapsto [m \oplus c]]$ for some $m \in s(p)$. Again $\mathcal{F}(s) = N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s)$ and similarly to the case above

$$\begin{aligned} \mathcal{F}(s) &\xrightarrow{\epsilon}_G^* N_r^c N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &\xrightarrow{\epsilon}_G (c_p ! \text{msg}_{p',c}) \cdot (c_p ? \bullet) N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &\xrightarrow{\epsilon}_G (c_p ? \bullet) N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \oplus [\text{msg}_{p',c}]^{c_p} \\ &= (c_p ? \bullet) N^{sim} \parallel \mathcal{F}^\Pi([p \mapsto [m]]) \parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \oplus [\text{msg}_{p',c}]^{c_p} \\ &= (c_p ? \bullet) N^{sim} \parallel N^p \tilde{\Gamma}(m \circ \zeta^{-1}) N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \oplus [\text{msg}_{p',c}]^{c_p} \end{aligned}$$

It is immediate that we can derive

$$N^c \rightarrow^* \left(c_{\zeta(p_{(1)}^{col})} ! \bullet \right)^{|c(p_{(1)}^{col})|} \dots \left(c_{\zeta(p_{(n_{col})}^{col})} ! \bullet \right)^{|c(p_{(n_{col})}^{col})|} =: w$$

and that the equality $\mathbb{M}(w) = \tilde{\Gamma}(c \circ \zeta^{-1})$ holds. Thus

$$\begin{aligned} \mathcal{F}(s) &\xrightarrow{\epsilon}_G^* (c_p ? \bullet) N^{sim} \\ &\parallel (c_p ? \text{msg}_{p',c}) \cdot (c_p ! \bullet) \cdot N^{p'} \tilde{\Gamma}(m \circ \zeta^{-1}) \oplus \tilde{\Gamma}(c \circ \zeta^{-1}) N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \oplus [\text{msg}_{p',c}]^{c_p} \\ &= (c_p ? \bullet) N^{sim} \\ &\parallel (c_p ? \text{msg}_{p',c}) \cdot (c_p ! \bullet) \cdot N^{p'} \tilde{\Gamma}((m \oplus c) \circ \zeta^{-1}) N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \oplus [\text{msg}_{p',c}]^{c_p} \\ &\xrightarrow{\alpha}_G^* N^{sim} \parallel N^{p'} \left((m \oplus c) \circ \zeta^{-1} \circ \tilde{\Gamma} \right) N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &= N^{sim} \parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]] \oplus [p' \mapsto [m \oplus c]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \end{aligned}$$

Lastly, we clearly have

$$\begin{aligned} &\mathcal{F}^\Pi(s \ominus [p \mapsto [m]] \oplus [p' \mapsto [m \oplus c]]) \\ &= \mathcal{F}^\Pi(s \ominus [p \mapsto [m]] \oplus I \oplus [p' \mapsto [m \oplus c]] \oplus O) = \mathcal{F}^\Pi(s') \\ &\mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &= \mathcal{F}^\Gamma(s \ominus [p \mapsto [m]] \oplus I \oplus [p' \mapsto [m \oplus c]] \oplus O) = \mathcal{F}^\Gamma(s') \end{aligned}$$

from which we can deduce $\mathcal{F}(s) \xrightarrow{\alpha}_G^* \mathcal{F}(s')$, $(s', \mathcal{F}(s')) \in R$ and clearly $\alpha = s'$ which is what we wanted to prove.

- *$r \in \text{TransferRules}$.*

Then $r = ((p, I), (p', \mathcal{P}^{col}, O))$ since r is total and $s' = (s \ominus I \oplus [p \mapsto [m]]) \oplus O \oplus [p' \mapsto [\emptyset]] \oplus (m \circ \zeta^{-1})$ for some $m \in s(p)$.

Analogously to the cases above:

$$\begin{aligned} \mathcal{F}(s) &\xrightarrow{\epsilon}_G^* N_r^c N^{sim} \parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &\xrightarrow{\epsilon}_G (c_p ! \text{msg}_{p,\downarrow}) \cdot (c_p ? \bullet) \cdot (c ? ! \text{msg}_{p',\emptyset}) \cdot (c ? ? \bullet) N^{sim} \\ &\parallel \mathcal{F}^\Pi(s) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &\xrightarrow{\epsilon}_G^* (c_p ! \text{msg}_{p,\downarrow}) \cdot (c_p ? \bullet) \cdot (c ? ! \text{msg}_{p',\emptyset}) \cdot (c ? ? \bullet) N^{sim} \\ &\parallel N^p \tilde{\Gamma}(m \circ \zeta^{-1}) N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &\xrightarrow{\epsilon}_G (c_p ! \text{msg}_{p,\downarrow}) \cdot (c_p ? \bullet) \cdot (c ? ! \text{msg}_{p',\emptyset}) \cdot (c ? ? \bullet) N^{sim} \\ &\parallel (c_p ? \text{msg}_{p',\downarrow}) \cdot (c_p ! \bullet) \tilde{\Gamma}(m \circ \zeta^{-1}) N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &\xrightarrow{\epsilon}_G^* (c ? ! \text{msg}_{p',\emptyset}) \cdot (c ? ? \bullet) N^{sim} \\ &\parallel \tilde{\Gamma}(m \circ \zeta^{-1}) N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \\ &\xrightarrow{\epsilon}_G (c ? ! \text{msg}_{p',\emptyset}) \cdot (c ? ? \bullet) N^{sim} \\ &\parallel N^{\nu?} \parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O) \oplus \Gamma(\tilde{\Gamma}(m \circ \zeta^{-1})) \end{aligned}$$

We can see that

$$\mathcal{F}^\Gamma(s \ominus I \oplus O) \oplus \Gamma(\tilde{\Gamma}(m \circ \zeta^{-1})) = \mathcal{F}^\Gamma(s \ominus I \oplus O \oplus m \circ \zeta^{-1})$$

and so

$$\begin{aligned} \mathcal{F}(s) &\xrightarrow{\epsilon}_G^* (c ? ! \text{msg}_{p',\emptyset}) \cdot (c ? ? \bullet) N^{sim} \\ &\parallel (c ? ? \text{msg}_{p,\emptyset}) \cdot (c ? ! \bullet) \cdot N^{p'} \cdot N^{\nu?} \\ &\parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O \oplus (m \circ \zeta^{-1})) \\ &\xrightarrow{\alpha}_G^* N^{sim} \parallel N^{p'} \cdot N^{\nu?} \parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O \oplus (m \circ \zeta^{-1})) \\ &= N^{sim} \parallel \mathcal{F}^\Pi(s \ominus [p \mapsto [m]] \oplus [p' \mapsto [\emptyset]]) \\ &\triangleleft \mathcal{F}^\Gamma(s \ominus I \oplus O \oplus (m \circ \zeta^{-1})) \end{aligned}$$

Analysing the last configuration:

$$\begin{aligned} &\mathcal{F}^\Pi(s \ominus [p \mapsto [m]] \oplus [p' \mapsto [\emptyset]]) \\ &= \mathcal{F}^\Pi(s \ominus [p \mapsto [m]] \oplus I \oplus [p' \mapsto [\emptyset]] \oplus O \oplus (m \circ \zeta^{-1})) \\ &= \mathcal{F}^\Pi(s') \\ &\mathcal{F}^\Gamma(s \ominus I \oplus O \oplus (m \circ \zeta^{-1})) \\ &= \mathcal{F}^\Gamma(s \ominus [p \mapsto [m]] \oplus I \oplus [p' \mapsto [\emptyset]] \oplus O \oplus (m \circ \zeta^{-1})) \\ &= \mathcal{F}^\Gamma(s') \end{aligned}$$

from which we can deduce $\mathcal{F}(s) \xrightarrow{\alpha}_G^* \mathcal{F}(s')$, $(s', \mathcal{F}(s')) \in R$ and clearly $\alpha = s'$ which is what we wanted to prove.

Hence we can conclude that R is a weak simulation.

Let us now investigate R^{-1} . Suppose we have $(s, \mathcal{F}(s)) \in R$ and $F(s) \xrightarrow{s'}^* t$ then from the labelling we know we have

$\mathcal{F}(s) \xrightarrow{s'}_{\mathcal{G}} \mathcal{F}(s') \xrightarrow{\epsilon}_{\mathcal{G}} t$. Inspecting the rules of \mathcal{G} we can see that the only paths (modulo different interleavings) to make the transitions $\mathcal{F}(s) \xrightarrow{s'}_{\mathcal{G}} \mathcal{F}(s')$ are the three described in the proof that R is a simulation and depends on which rule r of \mathcal{N} is simulated.

Inspecting the transition paths we can see that if $r \in \text{SimpleRules}$ then $s' = s \ominus I \oplus O$ and since N_r^I can be fully executed we can deduce that $|s(p)| \geq |I(p)|$ for all $p \in \mathcal{P}^S$ and hence $s \ominus I \in \text{Config}$. Thus we can conclude that $s \xrightarrow{r}_{\mathcal{N}} s'$ and hence $s \xrightarrow{s'}_{\mathcal{N}} s'$.

If $r \in \text{ComplexRules}$ then we can see that $(s \ominus I \oplus [p \mapsto [m]]) \oplus O \oplus [p' \mapsto [m \oplus c]]$ for some $m \in s(p)$ and as above since N_r^I can be fully executed we can deduce that $s \ominus I \in \text{Config}$ which implies $s \xrightarrow{r}_{\mathcal{N}} s'$ and hence $s \xrightarrow{s'}_{\mathcal{N}} s'$.

For the third case if $r \in \text{ComplexRules}$ inspecting the transition path above we can see that $s' = s \ominus [p \mapsto [m]] \ominus I \oplus [p' \mapsto [\emptyset]] \oplus O \oplus (m \circ \zeta^{-1})$ for some $m \in s(p)$ as above since N_r^I can be fully executed we can deduce that $s \ominus I \in \text{Config}$. Hence $s \xrightarrow{r}_{\mathcal{N}} s'$ and hence $s \xrightarrow{s'}_{\mathcal{N}} s'$.

Thus we can conclude that R^{-1} is a weak simulation and thus R is a weak bisimulation.

In order to finish the proof we will now prove that a check of coverability of s_{cov} for \mathcal{N} is equivalent to a program-point coverability check of l_{cov} for \mathcal{G} .

First suppose $(\mathcal{N}, s_0, s_{\text{cov}})$ is a yes-instance of NNCT coverability with a simple query. Hence there exists a path $s_0 \rightarrow_{\mathcal{N}} s_1 \rightarrow_{\mathcal{N}} \dots \rightarrow_{\mathcal{N}} s_n$ and $s_{\text{cov}} \leq_{\text{Config}} s_n$. Attaching the labels to the transition system as we have described above this path turns into $s_0 \xrightarrow{s_1}_{\mathcal{N}} s_1 \xrightarrow{s_2}_{\mathcal{N}} \dots \xrightarrow{s_n}_{\mathcal{N}} s_n$. Since R is a weak bisimulation we know that we can find a path $S \xrightarrow{s_0}_{\mathcal{G}} \mathcal{F}(s_0) \xrightarrow{s_1}_{\mathcal{G}} \mathcal{F}(s_1) \xrightarrow{s_2}_{\mathcal{G}} \dots \xrightarrow{s_n}_{\mathcal{G}} \mathcal{F}(s_n)$. We know that for all $p \in \mathcal{P}^S$ it is the case that $|s_{\text{cov}}(p)| \leq |s_n(p)|$ hence $s_n \ominus (s_{\text{cov}} \upharpoonright \mathcal{P}^S) \in \text{Config}^{APCPS}$. Thus

$$\begin{aligned} \mathcal{F}(s_n) &= N^{\text{sim}} \parallel \mathcal{F}^{\Pi}(s_n) \triangleleft \mathcal{F}^{\Gamma}(s_n) \\ &\xrightarrow{\epsilon}_{\mathcal{G}} (c_{p(1)} ? \bullet)^{|s_{\text{cov}}(p(1))|} \dots (c_{p(n_S)} ? \bullet)^{|s_{\text{cov}}(p(n_S))|} l_{\text{cov}} \\ &\quad \parallel \mathcal{F}^{\Pi}(s_n) \triangleleft \mathcal{F}^{\Gamma}(s_n) \\ &\xrightarrow{\epsilon}_{\mathcal{G}} l_{\text{cov}} \parallel \mathcal{F}^{\Pi}(s_n) \triangleleft \mathcal{F}^{\Gamma}(s_n) \ominus \left[\bullet^{|s_{\text{cov}}(p(1))|} \right]^{c_{p(1)}} \\ &\quad \ominus \left[\bullet^{|s_{\text{cov}}(p(2))|} \right]^{c_{p(2)}} \ominus \dots \ominus \left[\bullet^{|s_{\text{cov}}(p(n_S))|} \right]^{c_{p(n_S)}} \\ &= A_{\text{cov}} \parallel \mathcal{F}^{\Pi}(s_n) \triangleleft \mathcal{F}^{\Gamma}(s_n \ominus (s_{\text{cov}} \upharpoonright \mathcal{P}^S)) \end{aligned}$$

Hence $(\mathcal{G}, S \triangleleft \emptyset, A_{\text{cov}} \triangleleft \emptyset)$ is a yes-instance of alternative simple coverability and thus $(\mathcal{G}, S \triangleleft \emptyset, A_{\text{cov}} \triangleleft \emptyset)$ is a yes-instance of standard simple coverability.

Suppose $(\mathcal{G}, S \triangleleft \emptyset, A_{\text{cov}} \triangleleft \emptyset)$ is a yes-instance of standard simple coverability. Hence we know that $(\mathcal{G}, S \triangleleft \emptyset, A_{\text{cov}} \triangleleft \emptyset)$ is a yes-instance of alternative simple coverability. Hence $S \xrightarrow{s_0}_{\mathcal{G}} A_{\text{cov}} \alpha \parallel \Pi \triangleleft \Gamma$ for some α , Π and Γ . Let us view this path in the labelled transition system we can then obtain: $S \xrightarrow{s_0}_{\mathcal{G}} \mathcal{F}(s_0) \xrightarrow{s_1}_{\mathcal{G}} \mathcal{F}(s_1) \xrightarrow{s_2}_{\mathcal{G}} \dots \xrightarrow{s_n}_{\mathcal{G}} \mathcal{F}(s_n) \xrightarrow{\epsilon}_{\mathcal{G}} A_{\text{cov}} \alpha \parallel \Pi \triangleleft \Gamma$ where it is clear that the first simulation state set up by S can only by s_0 .

We know that $\mathcal{F}(s_n) = N^{\text{sim}} \parallel \mathcal{F}^{\Pi}(s_n) \triangleleft \mathcal{F}^{\Gamma}(s_n)$. Since on the sequential level it must be the case that $N^{\text{sim}} \rightarrow^* \beta A_{\text{cov}} \alpha$, but inspecting the rules of \mathcal{G} we can see that only $N^{\text{sim}} \rightarrow^* (c_{p(1)} ? \bullet)^{|s_{\text{cov}}(p(1))|} \dots (c_{p(n_S)} ? \bullet)^{|s_{\text{cov}}(p(n_S))|} A_{\text{cov}}$ we can conclude that $\alpha = \epsilon$. Further, along these transition no administrative messages to channels $c_?$, c_p for $p \in \mathcal{P}^C$ are sent and thus any process in $\mathcal{F}^{\Pi}(s_n)$ is blocked on its initial receive action and thus cannot receive or send any messages to a c_p where $p \in \mathcal{P}^S$. Hence it must be the case that

$$\begin{aligned} \Gamma &= \mathcal{F}^{\Gamma}(s_n) \ominus \left[\bullet^{|s_{\text{cov}}(p(1))|} \right]^{c_{p(1)}} \ominus \dots \ominus \left[\bullet^{|s_{\text{cov}}(p(n_S))|} \right]^{c_{p(n_S)}} \\ &= \mathcal{F}^{\Gamma}(s_n \ominus s_{\text{cov}}) \end{aligned}$$

since $s_{\text{cov}}(p) = \emptyset$ for all $p \in \mathcal{P}^C$. We can thus conclude that $s_{\text{cov}} \leq_{\text{Config}} s_n$. Since R is a weak bisimulation we know that there is a path $s_0 \xrightarrow{s_1}_{\mathcal{N}} s_1 \xrightarrow{s_2}_{\mathcal{N}} \dots \xrightarrow{s_n}_{\mathcal{N}} s_n$ and hence $s_0 \rightarrow_{\mathcal{N}}^* s_n$ and $s_{\text{cov}} \leq_{\text{Config}} s_n$ thus $(\mathcal{N}, s_0, s_{\text{cov}})$ is a yes-instance of coverability with a simple query.

For boundedness, suppose the set $\{\Pi \triangleleft \Gamma : S \triangleleft \emptyset \rightarrow_{\mathcal{G}}^* \Pi \triangleleft \Gamma\}$ is finite. This implies that the set $\{\mathcal{F}(s) : \mathcal{F}(s_0) \rightarrow_{\mathcal{G}}^* \mathcal{F}(s)\}$ is finite and since R is a weak bisimulation this means that $\{s : s_0 \triangleleft \triangleleft \rightarrow_{\mathcal{N}}^* s\}$ is finite. Conversely, suppose $\{s : s_0 \triangleleft \triangleleft \rightarrow_{\mathcal{N}}^* s\}$ is finite then clearly $\{\mathcal{F}(s) : \mathcal{F}(s_0) \rightarrow_{\mathcal{G}}^* \mathcal{F}(s)\}$ is finite. Since in \mathcal{G} there are only finitely many more steps along weak bisimulation steps we can also conclude that $\{\Pi \triangleleft \Gamma : S \triangleleft \emptyset \rightarrow_{\mathcal{G}}^* \Pi \triangleleft \Gamma\}$ is finite. Hence \mathcal{G} is bounded from from $S \triangleleft \emptyset$ if, and only if, \mathcal{N} is bounded from s_0 .

For termination, suppose there is an infinite path from $S \triangleleft \emptyset$ in \mathcal{G} , then since there are only finitely many ϵ steps along weak bisimulation steps and R is a weak bisimulation we can conclude that \mathcal{N} has an infinite path from s_0 . Conversely, if \mathcal{N} has an infinite path from s_0 then there is an infinite path from $S \triangleleft \emptyset$ in \mathcal{G} since R is a weak bisimulation and every step \mathcal{N} is bounded from s_0 . Hence \mathcal{G} is terminating from from $S \triangleleft \emptyset$ if, and only if, \mathcal{N} is terminating from s_0 . \square

C. Proofs Upper Bound

Lemma 3. Suppose s is a covering path for s_{cov} in \mathcal{N}_{i+1} and $|s(1)(p_{i+1})| \geq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$. Then $\exists s'$ a covering path for s_{cov} in \mathcal{N}_{i+1} such that $s'(1) = s(1)$ and $|s'| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}})$.

Proof. Let $m_\infty = [\bullet \mapsto \infty]$, let $s = s(1)$ and define $s_\infty(i) = s(j)[p_{i+1} \mapsto m_\infty]$ for all $1 \leq j \leq |s|$. Since s is covering path for s_{cov} in \mathcal{N}_{i+1} , we can easily see that s_∞ is a covering path for s_{cov} in \mathcal{N}_i . Hence there must exists a covering path s'_∞ for s_{cov} in \mathcal{N}_i such that $|s'_\infty| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}})$. We can “replay” the path s'_∞ from s instead of $s[p_{i+1} \mapsto m_\infty]$ to obtain a configuration sequence s_0 from s so that $s_0(j)(k) = s'_\infty(j)(k)$ for all $1 \leq j \leq |s'_\infty|$ and $k \neq i+1$. An easy induction shows that for all $1 \leq j \leq |s'_\infty|$ we have $|s_0(j)(i+1)| \geq R' \times \rho_{\mathcal{N}_i}(s_{\text{cov}}) - j \times R$ from which we can conclude

$$\begin{aligned} |s_0(j)(i+1)| &\geq R' - R + (R' - R) \times (\rho_{\mathcal{N}_i}(s_{\text{cov}}) - 1) \\ &\geq R' - R \geq s_{\text{cov}}(p_{i+1}). \end{aligned}$$

since $\rho_{\mathcal{N}_i}(s_{\text{cov}}) \geq 1$ for all i . Hence we can conclude that s_0 is a path in \mathcal{N}_{i+1} , further s_0 is a path from s covering s_{cov} in \mathcal{N}_{i+1} and by construction $|s_0| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}})$. which is what we wanted to prove. \square

Lemma 4. For all covering paths s for s_{cov} one of two cases applies: (C-1) $\|s\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$; or (C-2) $s = s_1 \cdot s_p \cdot s_2$ such that $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$ and $\|s_p\|_{\mathcal{N}_{i+1}} \geq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$.

Proof. Let s be a covering path s for s_{cov} . Let us do a case analysis:

(i) Case 1: $\|s\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$.
Case C-1 holds.

(ii) Case 2: $\|s\|_{\mathcal{N}_{i+1}}^* \not\leq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$.
Hence we can split s in two paths $s = s_1 s_0$ such that s_0 is the longest prefix of s such that $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$ and $\|s_0(1)\|_{\mathcal{N}_{i+1}} \not\leq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$, i.e. $\|s_0(1)\|_{\mathcal{N}_{i+1}} \geq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$. Letting $s_p = s_0(1)$ and $s_2 = s_0(2) \cdots s_0(|s_0|)$ means C-2 applies. \square

Lemma 5. Suppose s is a covering path for s_{cov} in \mathcal{N}_{i+1} . If $L \in \mathbb{N}$ such that $|s| \leq L$ and $\|s(1)\|_{\mathcal{N}_{i+1}} < L \cdot R'$ then there exists a covering path s'' for s_{cov} in \mathcal{N}_{i+1} such that $s''(1) \leq_{\text{Config}} s(1)$, $\|s''(1)\|_{\mathcal{N}_{i+1};C} < L \cdot R'$, and $|s''| \leq L$.

Proof. We will construct a path removing all superfluous complex tokens first. Suppose we label every complex token in $s(1)$ as “not-moved” and along s if a complex token is moved we change the label to “moved”. Since $|s| \leq L$ and each transition can move at most one complex token, and remove R complex empty token it must be the case that for each $p \in \mathcal{P}^C$ the size of the multiset $M_{\text{remove}} = |[m \text{ “not-moved”} \mid m \in s(|s|)(p)]| \geq |[m \text{ “not-moved”} \mid m \in s(1)(p)]| - R \times L$. Hence the complex tokens in M_{remove} play no rôle in s , appear in $s(j)(p)$ for all $1 \leq j \leq |s|$

and could thus safely be remove without affecting the validity of the path. However, in $s(|s|)(p)$ there are complex tokens, at most $\|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C}$ many, that are required to justify $s_{\text{cov}} \leq_{\text{Config}} s(|s|)$ and hence we cannot remove these tokens without losing the cover of s_{cov} . We can be on the safe side and assume the former tokens are all “not-moved” tokens and hence we may only remove $|s(1)(p)| - (R \times L + \|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C})$ tokens. We can do this for all $p \in \mathcal{P}^C$ and hence we get a new path s''_0 which is a covering path for s_{cov} in \mathcal{N}_{i+1} , $|s''_0| \leq L$ and $|s''_0(1)(p')| < R \times L + \|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C}$ for all $p' \in \mathcal{P}^C$ and since $s''_0(1)$ is obtained from $s(1)$ by removing tokens we can see that $s''_0(1) \leq_{\text{Config}} s(1)$ and hence also $\|s''_0(1)\|_{\mathcal{N}_{i+1}} < (L+1) \cdot R'$.

Let us now show that we can also reduce the number of coloured tokens. We will label coloured tokens in $s''_0(1)$ with three statuses *untouched*, *ejected* and *consumed*. Initially this record of $s''_0(1)$ ’s coloured tokens marks all of them as *untouched*. Along each transition of s''_0 we update this record as follows. If the transition $s''_0(j) \rightarrow_{\mathcal{N}_{i+1}} s''_0(j+1)$ is a transfer transition then we look at all the coloured tokens that are ejected in this transition to simple places and label the coloured tokens whose origin lies in $s''_0(1)$ as *ejected*. If in the transition $s''_0(j) \rightarrow_{\mathcal{N}_{i+1}} s''_0(j+1)$ a simple token is removed and its origin is as a coloured token in $s''_0(1)$ then we mark that coloured token in $s''_0(1)$ as *consumed*. Since $|s| \leq L$ and each transition can remove at most R simple tokens it must be the case that at most $R \times L$ coloured tokens were marked as *consumed* by the above process.

However, for each $p' \in \mathcal{P}^C$ and $p^{\text{col}} \in \mathcal{P}^{\text{col}}$ there are coloured tokens in $s''_0(|s''_0|)(p')(p^{\text{col}})$, at most $\|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C}$ many, that are required to justify $s_{\text{cov}} \leq_{\text{Config}} s''_0(|s''_0|)$ and hence we cannot remove these tokens without losing the cover of s_{cov} . As above it is safe to label these coloured tokens *consumed* in $s''_0(1)$. Hence for each $p' \in \mathcal{P}^C$ and $p^{\text{col}} \in \mathcal{P}^{\text{col}}$ there are at most $R \times L + \|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C}$ that are labelled *consumed* in $s''_0(1)$.

It should be clear that we can remove all coloured tokens labelled *untouched* along s''_0 without affecting the validity of the path. We can also remove coloured tokens that are labelled *ejected* along s''_0 since the transfer transition may still fire, however, fewer tokens will be transferred. Since these coloured tokens are not labelled as *consumed* it is clear that their absence as simple tokens cannot disable a rule that is fired along s''_0 . For a rule r that removes an empty complex token m we can see that if m is empty along s''_0 then removing coloured tokens from $s''_0(1)$ could only possibly lead to less tokens inside m , hence it is impossible to disable r . Thus we can obtain a new covering path s'' for s_{cov} in \mathcal{N}_{i+1} , such that $|s''| \leq L$ and $|s''(1)(p')| < R \times L + \|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C}$ for all $p' \in \mathcal{P}^C$ and $|s''(1)(p')(p^{\text{col}})| < R \times L + \|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C}$ for all $p' \in \mathcal{P}^C$, for all $p^{\text{col}} \in \mathcal{P}^{\text{col}}$ since $s''(1)$ is obtained from $s(1)$ by removing tokens we can see that $s''(1) \leq_{\text{Config}} s(1)$ and hence also $\|s''_0(1)\|_{\mathcal{N}_{i+1}} < L \cdot R'$.

Since

$$R \times L + \|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C} \leq R + R \times (L - 1) + \|s_{\text{cov}}\|_{\mathcal{N}_{n_s};C} + 1$$

$$\begin{aligned}
&= R \times (L - 1) + R' \\
&\leq R' \times (L - 1) + R' \\
&= R' \times L
\end{aligned}$$

we can see that the above implies that $\|s_0''(1)\|_{\mathcal{N}_{i+1};C} < L \cdot R'$ which is what we wanted to prove. \square

Corollary 1. *For all covering paths s for s_{cov} in \mathcal{N}_{i+1} one of two cases applies:*

- (C'_1) $\|s\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$; or
- (C'_2) *there exist paths s_1 and $s_{p'} \cdot s_2$ such that s_1 is a covering path for $s_{p'}$, $s_1(1) = s(1)$ and $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$; $s_{p'} \cdot s_2$ is a covering path for s_{cov} and $|s_2'| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}})$; and $\|s_{p'}\|_{\mathcal{N}_{i+1};C} \leq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$.*

Proof. Let s be a covering path s for s_{cov} in \mathcal{N}_{i+1} . According to Lemma 4 we can consider the following two cases:

- (i) $\|s\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$.
Clearly case (C'_1) applies.
- (ii) $s = s_1 \cdot s_p \cdot s_2$ such that $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$ and $\|s_p\|_{\mathcal{N}_{i+1}} \geq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$.
Since $\|s_p\|_{\mathcal{N}_{i+1}} \geq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$ we can assume w.l.o.g. that $|s_p(p(i+1))| \geq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$ (we can always change the enumeration \mathcal{P}^s 's elements). Lemma 3 applies to $s_p \cdot s_2$ and gives us a covering path s_2' for s_{cov} from s_p in \mathcal{N}_{i+1} such that $|s_2'| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}})$.
Let $s_0 = s_1(|s_1|)$. Clearly $s_0 \cdot s_2'$ is a covering path for s_{cov} in \mathcal{N}_{i+1} , with length $|s_0 \cdot s_2'| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}}) + 1$, and $\|s_0\|_{\mathcal{N}_{i+1}} < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$ since $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$. We are thus able to apply Lemma 5 to $s_0 \cdot s_2'$ which yields a new covering path $s_{p'} \cdot s_2''$ for s_{cov} in \mathcal{N}_{i+1} such that $s_{p'} \leq_{\text{Config}} s_0$, $\|s_{p'}\|_{\mathcal{N}_{i+1};C} < R' \cdot (\rho_{\mathcal{N}_i}(s_{\text{cov}}) + 1)$, and $|s_{p'} \cdot s_2''| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}}) + 1$.
We can then conclude that s_1 is a covering path in \mathcal{N}_{i+1} for $s_{p'}$, $s_1(1) = s(1)$, and $\|s_1\|_{\mathcal{N}_{i+1}}^* < R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$. Further $s_{p'} \cdot s_2''$ is a covering path for s_{cov} in \mathcal{N}_{i+1} , $|s_2''| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}})$, and $\|s_{p'}\|_{\mathcal{N}_{i+1};C} \leq R' \cdot \rho_{\mathcal{N}_i}(s_{\text{cov}})$. \square

Proposition 4. (i) $\rho_{\mathcal{N}_0}(s_{\text{cov}}) \leq d_{\mathcal{N}_0}(S_{(0,R')})$
(ii) $\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)}) + \rho_{\mathcal{N}_i}(s_{\text{cov}})$.

Proof. In order to prove (i) let s be a covering path for s_{cov} in \mathcal{N}_0 . Since all simple places are ignored it is trivial to see that $\|s\|_{\mathcal{N}_0}^* < 1 \leq R'$. Further $\|s_{\text{cov}}\|_{\mathcal{N}_0;C} \leq \|s_{\text{cov}}\|_{\mathcal{N}_s;C} \leq R'$. Hence we have $s(1), s_{\text{cov}} \in S_{(0,R')}$ which implies we can find a path s' from $s(1)$ covering s_{cov} in \mathcal{N}_0 with $|s'| \leq d_{\mathcal{N}_0}(S_{(0,R')})$. Hence $\text{dist}_{\mathcal{N}_0}(s(1), s_{\text{cov}}) \leq d_{\mathcal{N}_0}(S_{(0,R')})$. Since s was arbitrary we can conclude that $\rho_{\mathcal{N}_0}(s_{\text{cov}}) \leq d_{\mathcal{N}_0}(S_{(0,R')})$.

For claim (ii) let s be an element of Config^∞ such that $\|s\|_{\mathcal{N}_{i+1}} < \infty$. We want to show that

$$\text{dist}_{\mathcal{N}_{i+1}}(s, s_{\text{cov}}) \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)}) + \rho_{\mathcal{N}_i}(s_{\text{cov}}).$$

Let us do a case analysis:

- (i) *Case 1: There is no covering path for s_{cov} from s in \mathcal{N}_{i+1} . Then $\text{dist}_{\mathcal{N}_{i+1}}(s, s_{\text{cov}}) = 0$ by definition and the inequality holds trivially.*
- (ii) *Case 2: There is a covering path s for s_{cov} from s in \mathcal{N}_{i+1} .*

Corollary 1 allows us to consider two cases:

Case (A): $\|s'\|_{\mathcal{N}_{i+1}}^ < B_i$.*

In this case it is easy to see that $\|s_{\text{cov}}\|_{\mathcal{N}_{i+1};C} \leq \|s_{\text{cov}}\|_{\mathcal{N}_s;C} \leq R' \leq B_i$. Hence we can see $s, s_{\text{cov}} \in S_{(i+1,B_i)}$ which implies we can find a path s' from s covering s_{cov} in \mathcal{N}_{i+1} with $|s'| \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)})$. Hence $\text{dist}_{\mathcal{N}_{i+1}}(s, s_{\text{cov}}) \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)})$.

Case (B): There exist paths s_1 and $s_{p'} \cdot s_2$ such that s_1 is a covering path for $s_{p'}$, $s_1(1) = s$ and $\|s_1\|_{\mathcal{N}_{i+1}}^ < B_i$; $s_{p'} \cdot s_2$ is a covering path for s_{cov} and $|s_2'| \leq \rho_{\mathcal{N}_i}(s_{\text{cov}})$; and $\|s_{p'}\|_{\mathcal{N}_{i+1};C} \leq B_i$.*

We can see $s, s_{p'} \in S_{(i+1,B_i)}$ which implies we can find a path s' from s covering $s_{p'}$ in \mathcal{N}_{i+1} with $|s'| \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)})$. Since \mathcal{N}_{i+1} is a WSTS and $s_{p'} \leq_{\text{Config}} s'(|s'|)$, we can replay $s_{p'} \cdot s_2$ from $s'(|s'|)$ yielding a path $s'(|s'|) \cdot s''$ in \mathcal{N}_{i+1} such that $|s''| = |s_2|$ and s'' covers s_{cov} . Thus $s' s''$ is a covering path in \mathcal{N}_{i+1} for s_{cov} with $|s' s''| \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)}) + \rho_{\mathcal{N}_i}(s_{\text{cov}})$. Hence we can conclude that $\text{dist}_{\mathcal{N}_{i+1}}(s, s_{\text{cov}}) \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)}) + \rho_{\mathcal{N}_i}(s_{\text{cov}})$.

Since the inequality holds in all cases and s was arbitrary we can conclude that

$$\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) \leq d_{\mathcal{N}_{i+1}}(S_{(i+1,B_i)}) + \rho_{\mathcal{N}_i}(s_{\text{cov}}).$$

\square

1) *Proof of Theorem 5:* Our analysis of $S_{(i,B)}$ is driven by two observations: (O'_1) $S_{(i,B)}$ is the set of starting and covering configurations of paths in \mathcal{N}_i along which the i not ignored places contain less than B tokens; (O'_2) along such paths a complex token cannot carry more than B tokens of a particular colour if these coloured tokens are ejected on the path; and if they are not ejected then they play no rôle in enabling the path. In order to exploit these observations we derive a transition system from \mathcal{N}_i that generates exactly the paths of bounded norm. Suppose $(S, \rightarrow_S, \leq_S)$ is a PSTS and $\|\cdot\|$ is a norm for S then we say $\mathcal{S} = (S, \rightarrow_S, \leq_S, \|\cdot\|)$ is a *normed PSTS*. Given a normed PSTS $\mathcal{S} = (S, \rightarrow_S, \leq_S, \|\cdot\|)$ and a *norm-radius* $\rho \in \mathbb{N}^\infty$ we define a normed sub-PSTS $\mathbb{B}_\rho[S]$ — the *ball* of norm-radius ρ in \mathcal{S} — by $\mathbb{B}_\rho[S] = (\mathbb{B}_\rho[S], \rightarrow_{\mathbb{B}_\rho[S]}, \leq_S, \|\cdot\|)$ where $\mathbb{B}_\rho[S] := \{s \in S \mid \|s\| < \rho\}$, and $\rightarrow_{\mathbb{B}_\rho[S]} := \rightarrow_S \cap (\mathbb{B}_\rho[S])^2$. Note that by construction all $\mathbb{B}_\rho[S]$ -paths s satisfy $\|s\|^* < \rho$.

Attaching the norm $\|\cdot\|_{\mathcal{N}_i}$ to \mathcal{N}_i we then obtain a normed PSTS $(\text{Config}^\infty, \rightarrow_{\mathcal{N}_i}, \leq_{\text{Config}}, \|\cdot\|_{\mathcal{N}_i})$ to which we will also refer to as \mathcal{N}_i . The transition system $\mathbb{B}_B[\mathcal{N}_i]$ produces the paths in $P_{i,B}$ which generate the set $S_{(i,B)}$.

Let $B \in \mathbb{N}$ and $i \leq n_S$, we will now define the Petri net $\mathcal{V}_{i,B}$. It is our intention to show there is a tight (length preserving) correspondence between $\mathbb{B}_B[\mathcal{N}_i]$ -paths and paths in $\mathcal{V}_{i,B}$. Set the dimension $\widehat{d}_{i,B} := i + (B+1)((n_C+1)(B+1)^{n_{\text{col}}-1} - 1)$.

We define the counter abstraction function $\alpha_{i,B} : \text{Config}^\infty \rightarrow \mathbb{N}^{\widehat{d}_{i,B}}$ as $\alpha_{i,B}(s)(j) := |s(p_{(j)})|$ for $1 \leq j \leq i$ and

$$\alpha_{i,B}(s)(i + \theta) := \sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} s(p'_{(j_{n_{\text{col}}})})(m^{\text{col}})$$

where $\theta = \sum_{k=0}^{n_{\text{col}}} j_k \cdot (B+1)^k$, for all $1 \leq j_{n_{\text{col}}} \leq n_C$ and $0 \leq j_k \leq B$, $k = 0, \dots, n_{\text{col}} - 1$, and we write

$$\gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1}) = \{m \in \mathcal{M}^{\text{col}} \mid \min(m(p_k^{\text{col}}), B) = j_{k-1}\}.$$

Let us establish a “linearity” property of $\alpha_{i,B}$ to simplify proofs.

Lemma 16. Let $i \leq n_S$, $B \in \mathbb{N}$. Suppose $s, s' \in \text{Config}^\infty$ such that both $\|s\|_{\mathcal{N}_i}, \|s'\|_{\mathcal{N}_i} < \infty$, then $\alpha_{i,B}(s \oplus s') = \alpha_{i,B}(s) + \alpha_{i,B}(s')$ and if $s \ominus s' \in \text{Config}^\infty$ then $\alpha_{i,B}(s \ominus s') = \alpha_{i,B}(s) - \alpha_{i,B}(s')$.

Proof. It is clear that for all $1 \leq j \leq i$

$$\begin{aligned} \alpha_{i,B}(s \oplus s')(j) &= |(s \oplus s')(p_{(j)})| = (s \oplus s')(p_{(j)})(\bullet) \\ &= s(p_{(j)})(\bullet) + s'(p_{(j)})(\bullet) \\ &= |s(p_{(j)})| + |s'(p_{(j)})| \\ &= \alpha_{i,B}(s)(j) + \alpha_{i,B}(s')(j) \end{aligned}$$

and since $s \ominus s' \in \text{Config}^\infty$ we know $|s(p_{(j)})| - |s'(p_{(j)})| \geq 0$ and hence

$$\begin{aligned} \alpha_{i,B}(s \ominus s')(j) &= |(s \ominus s')(p_{(j)})| = (s \ominus s')(p_{(j)})(\bullet) \\ &= s(p_{(j)})(\bullet) - s'(p_{(j)})(\bullet) \\ &= |s(p_{(j)})| - |s'(p_{(j)})| \end{aligned}$$

$$= \alpha_{i,B}(s)(j) - \alpha_{i,B}(s')(j)$$

Further let $1 \leq j_{n_{\text{col}}} \leq n_C$ and $(j_0, \dots, j_{n_{\text{col}}-1}) \in \mathbb{N}^{\leq B}$. Then

$$\begin{aligned} \alpha_{i,B}(s \oplus s') &\left(i + \sum_{k=0}^{n_{\text{col}}} j_k \cdot (B+1)^k \right) \\ &= \sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} (s \oplus s')(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \\ &= \sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} \left(s(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) + s'(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \right) \\ &= \left(\sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} s(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \right) \\ &\quad + \left(\sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} s'(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \right) \\ &= \alpha_{i,B}(s) \left(i + \sum_{k=0}^{n_{\text{col}}} j_k \cdot (B+1)^k \right) \\ &\quad + \alpha_{i,B}(s') \left(i + \sum_{k=0}^{n_{\text{col}}} j_k \cdot (B+1)^k \right) \end{aligned}$$

and since $s \ominus s' \in \text{Config}^\infty$ we know $s(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) - s'(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \geq 0$ and hence

$$\begin{aligned} \alpha_{i,B}(s \ominus s') &\left(i + \sum_{k=0}^{n_{\text{col}}} j_k \cdot (B+1)^k \right) \\ &= \sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} (s \ominus s')(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \\ &= \sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} \left(s(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) - s'(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \right) \\ &= \left(\sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} s(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \right) \\ &\quad - \left(\sum_{m^{\text{col}} \in \gamma_B^{\text{col}}(j_0, \dots, j_{n_{\text{col}}-1})} s'(p'_{(j_{n_{\text{col}}})})(m^{\text{col}}) \right) \\ &= \alpha_{i,B}(s) \left(i + \sum_{k=0}^{n_{\text{col}}} j_k \cdot (B+1)^k \right) \\ &\quad - \alpha_{i,B}(s') \left(i + \sum_{k=0}^{n_{\text{col}}} j_k \cdot (B+1)^k \right) \end{aligned}$$

Hence we know that for all $1 \leq j \leq \widehat{d}_{i,B}$ $\alpha_{i,B}(s \oplus s')(j) = \alpha_{i,B}(s)(j) + \alpha_{i,B}(s')(j)$ and thus $\alpha_{i,B}(s \oplus s') = \alpha_{i,B}(s) + \alpha_{i,B}(s')$, and if $s \ominus s' \in \text{Config}^\infty$ then for $1 \leq j \leq \widehat{d}_{i,B}$ $\alpha_{i,B}(s \ominus s')(j) = \alpha_{i,B}(s)(j) - \alpha_{i,B}(s')(j)$ and thus

$$\alpha_{i,B}(s \ominus s') = \alpha_{i,B}(s) - \alpha_{i,B}(s').$$

□

We define the counter abstraction Petri net $\mathcal{V}_{i,B} = (\widehat{d}_{i,B},$

$\hat{F}_{i,B}$). The set of rules $\hat{F}_{i,B}$ is derived from \mathcal{R} by case analysis: if $r = (I, O) \in \text{SimpleRules}$ then add $\hat{r} := (\alpha_{i,B}(O \oplus I), \alpha_{i,B}(I))$ to $\hat{F}_{i,B}$.

For $r = ((p, I), (p', c, O)) \in \text{ComplexRules}$, add for all $m \in \mathcal{M}^{col}$ such that $\|m\|_{col} \leq B$ the rule $\hat{r}_m := (\hat{r}, d)$ to $\hat{F}_{i,B}$ where

$$\begin{aligned}\hat{r} &= \alpha_{i,B}(O \oplus [p' \mapsto [m \oplus c]] \oplus (I \oplus [p \mapsto [m]])) \\ d &= \alpha_{i,B}(I \oplus [p \mapsto [m]]).\end{aligned}$$

For $r = ((p, I), (p', P, O)) \in \text{TransferRules}$, define a collection of rules parametrised for all $m \in \mathcal{M}^{col}$ such that $\|m\|_{col} \leq B$ and $\max_{p \in P}(|m(p)|) < B$ as $\hat{r}_m := (\hat{r}, d)$ and add them to $\hat{F}_{i,B}$ where $m_P = m \upharpoonright P$ and $m_{\bar{P}} = m \upharpoonright (\mathcal{P}^{col} \setminus P)$:

$$\begin{aligned}\hat{r} &= \alpha_{i,B}(O \oplus (m_P \circ \zeta^{-1}) \oplus [p' \mapsto [m_{\bar{P}}]] \oplus (I \oplus [p \mapsto [m]])) \\ d &= \alpha_{i,B}(I \oplus [p \mapsto [m]]).\end{aligned}$$

Coverability instances for $\mathcal{V}_{i,B}$ and $\mathbb{B}_B[\mathcal{N}_i]$ are unfortunately not isomorphic since the order on the former is ignorant of complex tokens. To remedy this we add rules to $\mathcal{V}_{i,B}$ which yields the Petri net $\mathcal{V}_{i,B}^{\leq} = (\hat{d}_{i,B}, \hat{F}_{i,B} \cup \hat{F}_{i,B}^{\leq})$. Let us write $s_{p,m} := [p \mapsto [m]]$ for $p \in \mathcal{P}^c$, $m \in \mathcal{M}^{col}$ and define

$$\hat{F}_{i,B}^{\leq} = \{(\alpha_{i,B}(s_{p,m'}) - \alpha_{i,B}(s_{p,m})), \alpha_{i,B}(s_{p,m'}) \mid \Xi\}$$

where $\Xi = m, m' \in \mathcal{M}_{n_{col}, B+1}^{col}, m' >_{\mathcal{M}^{col}} m, p \in \mathcal{P}^c$.

Remark. Even though the Petri nets $\mathcal{V}_{i,B}$ and $\mathcal{V}_{i,B}^{\leq}$ have a vast number of rules, the maximal entry in any rule is tightly controlled: $R \geq \max\{\hat{r}(i) \mid \hat{r} \in \hat{F}_{i,B} \cup \hat{F}_{i,B}^{\leq}\}$. We aim to exploit a result by Bonnet et al. that shows that covering radii in SIAN are sensitive to the maximal entry in any rule rather than their number.

We will now set out to show the connection between $\mathcal{V}_{i,B}$ and \mathcal{N}_i . First let us add a norm to $\mathcal{V}_{i,B}$: define $\|v\|_{\mathcal{V}_{i,B}} = \max_{j \in \langle i \rangle}(|v(j)|)$. In the following we prove two lemmas to help us prove a lockstep property (Lemma 19) between $\mathcal{V}_{i,B}$ and \mathcal{N}_i .

Lemma 17. Let $1 \leq i \leq n_s$, $B \in \mathbb{N}$. Suppose $p = p'_{(j_{n_{col}})} \in \mathcal{P}^c$, and $j_0, \dots, j_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $m, m' \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1}) \subseteq \mathcal{M}^{col}$ and $m'' \in \mathcal{M}^{col}$ then

$$\alpha_{i,B}([p \mapsto [m \oplus m'']]) = \alpha_{i,B}([p \mapsto [m' \oplus m'']])$$

Proof. Let $j'_0, \dots, j'_{n_{col}-1}$ such that $m \oplus m'' \in \gamma_B^{col}(j'_0, \dots, j'_{n_{col}-1})$ and write $j'_{n_{col}} = j_{n_{col}}$. Let us consider $m_0 \oplus c$. Let $1 \leq k \leq n_{col}$ and note that since $m, m' \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ we have $\min(m(p_{(k)}^{col}), B) = j_{k-1} = \min(m(p_{(k)}^{col}), B)$; and thus

$$\begin{aligned}j_{k-1} &= \min((m \oplus m'')(p_{(k)}^{col}), B) \\ &= \min(m(p_{(k)}^{col}) + m''(p_{(k)}^{col}), B) \\ &= \min(\min(m(p_{(k)}^{col}), B) + m''(p_{(k)}^{col}), B) \\ &= \min(\min(m'(p_{(k)}^{col}), B) + m''(p_{(k)}^{col}), B) \\ &= \min(m'(p_{(k)}^{col}) + m''(p_{(k)}^{col}), B)\end{aligned}$$

from which we can conclude that $m' \oplus m'' \in \gamma_B^{col}(j'_0, \dots, j'_{n_{col}-1})$. Further

$$\begin{aligned}\alpha_{i,B}([p \mapsto [m \oplus m'']]) &\left(i + \sum_{k=0}^{n_{col}} j'_k \cdot (B+1)^k\right) \\ &= \sum_{m^{col} \in \gamma_B^{col}(j'_0, \dots, j'_{n_{col}-1})} ([p \mapsto [m \oplus m'']]) (p) (m^{col}) \\ &= ([p \mapsto [m \oplus m'']]) (p) (m \oplus m'') \\ &= 1 = ([p \mapsto [m' \oplus m'']]) (p) (m \oplus m'') \\ &= \sum_{m^{col} \in \gamma_B^{col}(j'_0, \dots, j'_{n_{col}-1})} ([p \mapsto [m' \oplus m'']]) (p) (m^{col}) \\ &= \alpha_{i,B}([p \mapsto [m' \oplus m'']]) \left(i + \sum_{k=0}^{n_{col}} j'_k \cdot (B+1)^k\right)\end{aligned}$$

For any $p'_{(j''_{n_{col}})} \in \mathcal{P}^c$, and $j''_0, \dots, j''_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $i + \sum_{k=0}^{n_{col}} j''_k \cdot (B+1)^k \neq i + \sum_{k=0}^{n_{col}} j'_k \cdot (B+1)^k$ it is clear that $m \oplus m'', m' \oplus m'' \notin \gamma_B^{col}(j''_0, \dots, j''_{n_{col}-1})$ and so we have

$$\begin{aligned}\alpha_{i,B}([p \mapsto [m \oplus m'']]) &\left(i + \sum_{k=0}^{n_{col}} j''_k \cdot (B+1)^k\right) \\ &= \sum_{m^{col} \in \gamma_B^{col}(j''_0, \dots, j''_{n_{col}-1})} ([p \mapsto [m \oplus m'']]) (p'_{(j''_{n_{col}})}) (m^{col}) \\ &= \sum_{m^{col} \in \gamma_B^{col}(j''_0, \dots, j''_{n_{col}-1})} 0 = 0 \\ &= \sum_{m^{col} \in \gamma_B^{col}(j''_0, \dots, j''_{n_{col}-1})} ([p \mapsto [m' \oplus m'']]) (p'_{(j''_{n_{col}})}) (m^{col}) \\ &= \alpha_{i,B}([p \mapsto [m' \oplus m'']]) \left(i + \sum_{k=0}^{n_{col}} j''_k \cdot (B+1)^k\right)\end{aligned}$$

and also for all $1 \leq j \leq n_s$

$$\begin{aligned}\alpha_{i,B}([p \mapsto [m \oplus m'']]) (j) &= |([p \mapsto [m \oplus m'']]) (p_{(j)})| \\ &= 0 \\ &= |([p \mapsto [m' \oplus m'']]) (p_{(j)})| \\ &= \alpha_{i,B}([p \mapsto [m' \oplus m'']]) (j)\end{aligned}$$

Hence we can conclude that

$$\alpha_{i,B}([p \mapsto [m \oplus m'']]) = \alpha_{i,B}([p \mapsto [m' \oplus m'']])$$

□

Lemma 18. Let $1 \leq i \leq n_s$, $B \in \mathbb{N}$. Suppose $p = p'_{(j_{n_{col}})} \in \mathcal{P}^c$, and $j_0, \dots, j_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $m, m' \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1}) \subseteq \mathcal{M}^{col}$ and $P \subseteq \mathcal{P}^{col}$ then

$$\alpha_{i,B}([p \mapsto [m \upharpoonright \bar{P}]]) = \alpha_{i,B}([p \mapsto [m' \upharpoonright \bar{P}]])$$

where we write $\bar{P} = \mathcal{P}^{col} \setminus P$.

Proof. Let $j'_0, \dots, j'_{n_{col}-1}$ such that $m \upharpoonright \bar{P} \in \gamma_B^{col}(j'_0, \dots, j'_{n_{col}-1})$ and write $j'_{n_{col}} = j_{n_{col}}$. Let us consider $m' \upharpoonright \bar{P}$. Let $1 \leq k \leq n_{col}$ and note

that since $m, m' \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ we have $\min(m(p_{(k)}^{col}), B) = j_{k-1} = \min(m(p_{(k)}^{col}), B)$; and thus if $p_{(k)}^{col} \notin P$

$$\begin{aligned} j_{k-1} &= \min(m \upharpoonright \bar{P}(p_{(k)}^{col}), B) = \min(m(p_{(k)}^{col}), B) \\ &= \min(m'(p_{(k)}^{col}), B) \\ &= \min(m' \upharpoonright \bar{P}(p_{(k)}^{col}), B) \end{aligned}$$

and for $p_{(k)}^{col} \in P$

$$\begin{aligned} j_{k-1} &= \min(m \upharpoonright \bar{P}(p_{(k)}^{col}), B) = \min(0, B) = 0 = \\ &= \min(m' \upharpoonright \bar{P}(p_{(k)}^{col}), B) \end{aligned}$$

from which we can conclude that $m \upharpoonright \bar{P}, m' \upharpoonright \bar{P} \in \gamma_B^{col}(j'_0, \dots, j'_{n_{col}-1})$. Lemma 17 then gives

$$\alpha_{i,B}([p \mapsto [m \upharpoonright \bar{P}]]) = \alpha_{i,B}([p \mapsto [m' \upharpoonright \bar{P}]])$$

□

Lemma 19. Suppose $s \in Config^\infty$, $i \leq n_s$, $B \in \mathbb{N}$ such that $|s(p_{(j)})| = \infty$ for all $i < j \leq n_s$.

- (i) If $s \xrightarrow{\mathbb{B}_B[\mathcal{N}_i]} s'$ then we have $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s')$.
- (ii) If $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s) + r$ then there exists an $s' \in Config^\infty$ such that $\alpha_{i,B}(s') = \alpha_{i,B}(s) + r$ and $s \xrightarrow{\mathcal{N}_i} s'$.

Proof. We will first prove (i). Suppose $s \xrightarrow{\mathbb{B}_B[\mathcal{N}_i]} s'$ for some $r \in \mathcal{R}$ and let us write (a) $r = (I, O)$ if $r \in SimpleRules$; (b) $r = ((p, I), (p', c, O))$ if $r \in ComplexRules$; and (c) $r = ((p, I), (p', P, O))$ if $r \in TransferRules$.

Since $s \xrightarrow{\mathcal{N}_i} s'$ we know for all $p_{(j)} \in \mathcal{P}^s$ we have $|s(p_{(j)})| \geq |I(p_{(j)})|$ and for all $p' \in \mathcal{P}^c$ $s(p') \supseteq I(p')$. Hence $s \ominus I \in Config^\infty$ and $\alpha_{i,B}(s)(i) = |s(p_{(j)})| \geq |I(p_{(j)})| = \alpha_{i,B}(I)(j)$.

- *Case:* $s \xrightarrow{\mathcal{N}_i} s'$, $r = (I, O) \in SimpleRules$.

In this case we can see that for all $p'_{(j)} \in \mathcal{P}^c$

$$\begin{aligned} \alpha_{i,B}(s)(i + j(B+1)^{n_{col}}) &= s(p'_{(j)})(0) \\ &\geq I(p'_{(j)})(0) = \alpha_{i,B}(I)(i + j(B+1)^{n_{col}}). \end{aligned}$$

We see that $\alpha_{i,B}(r)$ is enabled at $\alpha_{i,B}(s)$ and hence there is a transition $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s) + \hat{r}$. We know that $s' = (s \ominus I) \oplus O$ so since $s \ominus I \in Config^\infty$ Lemma 16 gives us $\alpha_{i,B}(s') = (\alpha_{i,B}(s) - \alpha_{i,B}(I)) + \alpha_{i,B}(O)$ and clearly $\hat{r} = (-\alpha_{i,B}(I)) + \alpha_{i,B}(O)$ so $\alpha_{i,B}(s') = \alpha_{i,B}(s) + \hat{r}$ and thus $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s')$ which is what we wanted to prove.

- *Case:* $s \xrightarrow{\mathcal{N}_i} s'$, $r = ((p, I), (p', c, O)) \in ComplexRules$.

In this case we know that for some $m \in s(p)$ we have $s \ominus [p \mapsto [m]] \ominus I \in Config^\infty$ and

$$s' = (s \ominus [p \mapsto [m]] \ominus I) \oplus [p' \mapsto [m \oplus c]] \oplus O$$

Suppose $p = p'_{(j_{n_{col}})}$, and $j_0, \dots, j_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $m \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ then we know that

$$\alpha_{i,B}(s) \left(i + \sum_{k=0}^{n_{col}} j_k (B+1)^k \right)$$

$$= \sum_{m^{col} \in \gamma_B^{col}(j_0, \dots, j_{n_{col}})} s(p)(m^{col}) \geq s(p)(m) \geq 1.$$

We can thus see that for some $m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ such that $\|m_0\|_I \leq B$ the rule \hat{r}_{m_0} is enabled at $\alpha_{i,B}(s)$ and thus $\alpha_{i,B}(s) \xrightarrow{\hat{r}_{m_0}}_{\mathcal{V}_{i,B}} \alpha_{i,B}(s) + \hat{r}_{m_0}$. Since $s \ominus [p \mapsto [m]] \ominus I \in Config^\infty$ Lemma 16 gives us

$$\begin{aligned} \alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}([p \mapsto [m]]) \\ &\quad - \alpha_{i,B}(I) + \alpha_{i,B}([p' \mapsto [m \oplus c]]) + \alpha_{i,B}(O) \end{aligned}$$

Since $m, m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ Lemma 17 applies to give both

$$\begin{aligned} \alpha_{i,B}([p \mapsto [m]]) &= \alpha_{i,B}([p \mapsto [m_0]]) \\ \alpha_{i,B}([p' \mapsto [m \oplus c]]) &= \alpha_{i,B}([p' \mapsto [m_0 \oplus c]]) \end{aligned}$$

and hence

$$\begin{aligned} \alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}([p \mapsto [m_0]]) \\ &\quad - \alpha_{i,B}(I) + \alpha_{i,B}([p' \mapsto [m_0 \oplus c]]) + \alpha_{i,B}(O) \\ &= \alpha_{i,B}(s) + \hat{r}_{m_0} \end{aligned}$$

and thus $\alpha_{i,B}(s) \xrightarrow{\hat{r}_{m_0}}_{\mathcal{V}_{i,B}} \alpha_{i,B}(s')$ which is what we wanted to prove.

- *Case:* $s \xrightarrow{\mathcal{N}_i} s'$, $r = ((p, I), (p', P, O)) \in TransferRules$.

In this case we know that for some $m \in s(p)$ we have $s \ominus [p \mapsto [m]] \ominus I \in Config^\infty$ and

$$s' = (s \ominus [p \mapsto [m]] \ominus I) \oplus [p' \mapsto [m_{\bar{P}}]] \oplus O \oplus (m_P \circ \zeta^{-1})$$

where $m_P = m \upharpoonright P$ and $m_{\bar{P}} = m \upharpoonright \{\mathcal{P}^{col} \setminus P\}$. Since $\|s'\|_{\mathcal{N}_i} < B$ we can conclude that for all $p \in \zeta(P)$ we can bound $|(m_P \circ \zeta^{-1})(p)| < B$ which implies that

$$\max_{p \in P} (|m(p)|) < B.$$

Suppose $p = p'_{(j_{n_{col}})}$, and $j_0, \dots, j_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $m \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ then we know that

$$\begin{aligned} \alpha_{i,B}(s) \left(i + \sum_{k=0}^{n_{col}} j_k (B+1)^k \right) &= \sum_{m^{col} \in \gamma_B^{col}(j_0, \dots, j_{n_{col}})} s(p)(m^{col}) \\ &\geq s(p)(m) \geq 1. \end{aligned}$$

Further for all k such that $p_{(k)}^{col} \in P$ it is the case that $j_k < B$. Hence for some $m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ such that $\|m_0\|_{col} \leq B$ and $\max_{p \in P} (|m_0(p)|) < B$ the rule \hat{r}_{m_0} is enabled at $\alpha_{i,B}(s)$ and so $\alpha_{i,B}(s) \xrightarrow{\hat{r}_{m_0}}_{\mathcal{V}_{i,B}} \alpha_{i,B}(s) + \hat{r}_{m_0}$. Since $s \ominus [p \mapsto [m]] \ominus I \in Config^\infty$ Lemma 16 yields

$$\begin{aligned} \alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}([p \mapsto [m]]) - \alpha_{i,B}(I) \\ &\quad + \alpha_{i,B}([p' \mapsto [m_{\bar{P}}]]) \\ &\quad + \alpha_{i,B}(O) + \alpha_{i,B}((m_P \circ \zeta^{-1})). \end{aligned}$$

Since $m, m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ we can apply

Lemma 17 and Lemma 18 to get

$$\begin{aligned}\alpha_{i,B}([p \mapsto [m]]) &= \alpha_{i,B}([p \mapsto [m_0]]) \\ \alpha_{i,B}([p' \mapsto [m_{\overline{P}}]]) &= \alpha_{i,B}([p' \mapsto [m_0]_{\overline{P}}]])\end{aligned}$$

where $m_0_{\overline{P}} = m_0 \upharpoonright (\mathcal{P}^{col} \setminus P)$. Since $\max_{p \in P}(|m(p)|) < B$ and $\max_{p \in P}(|m_0(p)|) < B$ we can see that

$$\begin{aligned}m(p_{(k)}) &= \min(m(p_{(k)}), B) = j_{k-1} \\ &= \min(m_0(p_{(k)}), B) = m_0(p_{(k)})\end{aligned}$$

for all $1 \leq k \leq \mathcal{P}^{col}$ such that $p_{(k)}^{col} \in P$. Hence we can see that

$$m_P \circ \zeta^{-1} = m_{0P} \circ \zeta^{-1}$$

where $m_{0P} = m_0 \upharpoonright P$ which implies

$$\begin{aligned}\alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}([p \mapsto [m_0]]) - \alpha_{i,B}(I) \\ &\quad + \alpha_{i,B}([p' \mapsto [m_0]_{\overline{P}}]]) \\ &\quad + \alpha_{i,B}(O) + (m_{0P} \circ \zeta^{-1}) \\ &= \alpha_{i,B}(s) + \hat{r}_{m_0}\end{aligned}$$

and hence $\alpha_{i,B}(s) \xrightarrow{\hat{r}_{m_0}}_{\mathcal{V}_{i,B}} \alpha_{i,B}(s')$ which is what we wanted to prove.

Hence in all cases $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s')$ thus (i) holds.

For (ii) suppose $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s) + r$ for some $r \in \hat{F}_{i,B}$. We know that either (a) for some $r_0 = (I, O) \in SimpleRules$ we have $r = \hat{r}_{0}$; or (b) for some $m \in \mathcal{M}^{col}$ such that $\|m\|_{col} \leq B$, $r = \hat{r}_{0m}$ for some $r_0 = ((p, I), (p', c, O)) \in ComplexRules$; or (c) for some $m \in \mathcal{M}^{col}$ such that $\|m\|_{col} \leq B$ and for all $\max_{p \in P}(|m(p)|) < B$, $r = \hat{r}_{0m}$ for some $r_0 = ((p, I), (p', P, O)) \in TransferRules$.

Before we do a case analysis on r_0 let us have a look at the simple places. First of all $1 \leq j \leq i$, $p_{(j)} \in \mathcal{P}^S$ we know that $\alpha_{i,B}(s)(j) + r(j) \geq 0$, i.e. $|s(p_{(j)})| \geq |I(p_{(j)})|$ and we know that $|s(p_{(j)})| = \infty$ for $i < j \leq n_S$. Hence clearly $s \ominus (I \upharpoonright \mathcal{P}^S) \in Config^\infty$.

We proceed the proof by a case analysis on r .

- Case: $r = \hat{r}_0$ for $r_0 \in SimpleRules$.

In this case we can see that for all $p'_{(j)} \in \mathcal{P}^C$

$$\begin{aligned}s(p'_{(j)})(0) &= \alpha_{i,B}(s)(i + j(B+1)^{n_{col}}) \\ &\geq \alpha_{i,B}(I)(i + j(B+1)^{n_{col}}) = I(p'_{(j)})(0).\end{aligned}$$

Hence $s \ominus I \in Config^\infty$. Further we know that

$$\alpha_{i,B}(s) + r = \alpha_{i,B}(s) - \alpha_{i,B}(I) + \alpha_{i,B}(O)$$

and since by above $s \ominus I \in Config^\infty$ we get

$$\alpha_{i,B}(s) + r = \alpha_{i,B}(s \ominus I \oplus O)$$

if we define $s' = s \ominus I \oplus O$ then clearly $s \rightarrow_{\mathcal{N}_i} s'$ which is what we wanted to prove.

- Case: $r = \hat{r}_{0m}$ for $r_0 \in ComplexRules$.

Suppose $p = p'_{(j_{n_{col}})}$ and $j_0, \dots, j_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $m \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ then by definition $\hat{r}_{0m}(i + \sum_{k=0}^{n_{col}} j_k(B+1)^k) = -1$, since $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s) +$

r we must have that

$$\alpha_{i,B}(s)(\theta) + \hat{r}_{0m}(\theta) \geq 0$$

where $\theta = i + \sum_{k=0}^{n_{col}} j_k(B+1)^k$. Hence we can deduce that

$$\sum_{m^{col} \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})} |s(p'_{(j_{n_{col}})})(m^{col})| \geq 1$$

Hence there exists a $m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ such that $|s(p'_{(j_{n_{col}})})(m_0)| \geq 1$.

It is easy to see that $s \ominus I \ominus [p \mapsto [m_0]] \in Config^\infty$

$s \rightarrow_{\mathcal{N}_i} (s \ominus I \ominus [p \mapsto [m_0]]) \oplus O \oplus [p' \mapsto [m_0 \oplus c]] =: s'$

We can then use Lemma 16 to see that

$$\begin{aligned}\alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}(I) - \alpha_{i,B}([p \mapsto [m_0]]) \\ &\quad + \alpha_{i,B}(O) + \alpha_{i,B}([p' \mapsto [m_0 \oplus c]])\end{aligned}$$

Since $m, m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ and Lemma 17 yields

$$\begin{aligned}\alpha_{i,B}([p \mapsto [m_0]]) &= \alpha_{i,B}([p \mapsto [m]]) \\ \alpha_{i,B}([p' \mapsto [m_0 \oplus c]]) &= \alpha_{i,B}([p' \mapsto [m \oplus c]])\end{aligned}$$

Hence we can see that

$$\begin{aligned}\alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}(I) - \alpha_{i,B}([p \mapsto [m]]) \\ &\quad + \alpha_{i,B}(O) + \alpha_{i,B}([p' \mapsto [m \oplus c]]) \\ &= \alpha_{i,B}(s) + r\end{aligned}$$

which is what we wanted to prove.

- Case: $r = \hat{r}_{0m}$ for $r_0 \in TransferRules$.

Suppose $p = p'_{(j_{n_{col}})}$ and $j_0, \dots, j_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $m \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ then by definition $\hat{r}_{0m}(i + \sum_{k=0}^{n_{col}} j_k(B+1)^k) = -1$, since $\alpha_{i,B}(s) \rightarrow_{\mathcal{V}_{i,B}} \alpha_{i,B}(s) + r$ we must have that

$$\alpha_{i,B}(s)(\theta) + \hat{r}_{0m}(\theta) \geq 0.$$

where $\theta = i + \sum_{k=0}^{n_{col}} j_k(B+1)^k$. Hence we can deduce that

$$\sum_{m^{col} \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})} |s(p'_{(j_{n_{col}})})(m^{col})| \geq 1$$

Hence there exists a $m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ such that $|s(p'_{(j_{n_{col}})})(m_0)| \geq 1$.

Hence it is easy to see that $s \ominus I \ominus [p \mapsto [m_0]] \in Config^\infty$ and

$$s \rightarrow_{\mathcal{N}_i} s'$$

where

$$s' = s \ominus [p \mapsto [m_0]] \oplus I \oplus [p' \mapsto [m_0]_{\overline{P}}] \oplus O \oplus (m_{0P} \circ \zeta^{-1})$$

where $m_{0P} = m_0 \upharpoonright P$ and $m_{0\overline{P}} \upharpoonright (\mathcal{P}^{col} \setminus P)$. Since $s \ominus [p \mapsto [m_0]] \oplus I \in Config^\infty$ Lemma 16 yields

$$\begin{aligned}\alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}([p \mapsto [m_0]]) - \alpha_{i,B}(I) \\ &\quad + \alpha_{i,B}([p' \mapsto [m_0]_{\overline{P}}]]) + \alpha_{i,B}([p_0 \mapsto 0 \mid p_0 \in P])\end{aligned}$$

$$+ \alpha_{i,B}(O) + \alpha_{i,B}((m_0 \upharpoonright P) \circ \zeta^{-1} + \mathbf{0})).$$

Since $m, m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ we can apply Lemma 17 and Lemma 18 to get

$$\begin{aligned} \alpha_{i,B}([p \mapsto [m]]) &= \alpha_{i,B}([p \mapsto [m_0]]) \\ \alpha_{i,B}([p' \mapsto [m_{\overline{P}}]]) &= \alpha_{i,B}([p' \mapsto [m_0]_{\overline{P}}]]) \end{aligned}$$

where $m_{\overline{P}} = m \upharpoonright (\mathcal{P}^{col} \setminus P)$. Further we notice by assumption $\max p \in P | m(p) | < B$ and thus for all $1 \leq k \leq \mathcal{P}^{col}$ such that $p_{(k)}^{col} \in P$ we have

$$\begin{aligned} m(p_{(k)}) &= \min(m(p_{(k)}), B) = j_{k-1} \\ &= \min(m_0(p_{(k)}), B) = m_0(p_{(k)}) \end{aligned}$$

and hence we can see that

$$(m_P \circ \zeta^{-1}) = (m_{0P} \circ \zeta^{-1} + \mathbf{0})$$

where $m_P = m \upharpoonright P$ which implies

$$\begin{aligned} \alpha_{i,B}(s') &= \alpha_{i,B}(s) - \alpha_{i,B}([p \mapsto [m]]) - \alpha_{i,B}(I) \\ &\quad + \alpha_{i,B}([p' \mapsto [m_{\overline{P}}]]) \\ &\quad + \alpha_{i,B}(O) + \alpha_{i,B}(m_P \circ \zeta^{-1}) \\ &= \alpha_{i,B}(s') + r \end{aligned}$$

which is what we wanted to prove.

Thus in all cases also (ii) holds and hence we can conclude the proof. \square

Definition 8 (Simulation, bisimulation). Suppose (S, \xrightarrow{u}_S) and $(S', \xrightarrow{u}_{S'})$ are labelled transition systems we say a relation $\mathcal{B} \subseteq S \times S'$ is a (weak) simulation if for all $(s, s') \in \mathcal{B}$, if for some $t \in S$ we have $s \xrightarrow{u}_S t$ ($s \xrightarrow{u}_S^* t$) then there exists $t' \in S'$ such that $s' \xrightarrow{u}_{S'} t'$ ($s' \xrightarrow{u}_{S'}^* t'$) and $(t, t') \in \mathcal{B}$. We say \mathcal{B} is a bisimulation relation just if both \mathcal{B} and \mathcal{B}^{-1} are simulation relations.

Let us temporarily label the transition systems relations of \mathcal{N}_i , $\mathcal{V}_{i,B}$ in the following way: $s \xrightarrow{s'}_{\mathcal{N}_i} s'$ if $s \rightarrow_{\mathcal{N}_i} s'$; $v \xrightarrow{u}_{\mathcal{V}_{i,B}} v'$ if $v \rightarrow_{\mathcal{V}_{i,B}} v'$ where $u = s'$ if $\alpha_{i,B}(s') = v'$ and $u = \epsilon$ otherwise.

Proposition 5. The relation $\{(s, \alpha_{i,B}(s)) \mid s \in \mathbb{B}_B[Config^\infty], |s(p_{(j)})| = \infty, i < j\}$ is a bisimulation between the labelled transition systems $\mathbb{B}_B[\mathcal{N}_i]$ and $\mathbb{B}_B[\mathcal{V}_{i,B}]$.

Proof. Let $\mathcal{B} = \{(s, \alpha_{i,B}(s)) \mid s \in \mathbb{B}_B[Config^\infty], |s(p_{(j)})| = \infty, i < j\}$ and suppose $(s, \hat{s}) \in \mathcal{B}$. By definition we know that $\hat{s} = \alpha_{i,B}(s)$. Suppose $s \xrightarrow{s'}_{\mathbb{B}_B[\mathcal{N}_i]} s'$ is a transition then Lemma 19 yields that $\alpha_{i,B}(s) \xrightarrow{s'}_{\mathcal{V}_{i,B}} \alpha_{i,B}(s')$. Clearly $(s, \alpha_{i,B}(s')) \in \mathcal{B}$ since $s' \in \mathbb{B}_B[Config^\infty]$ which also implies that for all $1 \leq j \leq i$ we have $|\alpha_{i,B}(s')(j)| = |s'(p_{(j)})| < B$ and thus $\|\alpha_{i,B}(s')\|_{\mathcal{V}_{i,B}} < B$. Hence $\alpha_{i,B}(s) \xrightarrow{s'}_{\mathbb{B}_B[\mathcal{V}_{i,B}]} \mathcal{V}_{i,B} \alpha_{i,B}(s')$ is transition and so we can conclude that \mathcal{B} is a simulation relation.

Let us now consider \mathcal{B}^{-1} . Suppose $(s, \hat{s}) \in \mathcal{B}$, again we know that $\hat{s} = \alpha_{i,B}(s)$. Suppose $\alpha_{i,B}(s) \xrightarrow{u}_{\mathbb{B}_B[\mathcal{V}_{i,B}]} \alpha_{i,B}(s) + r$ is a transition for some $r \in R_{i,B}$. Lemma 19 then yields that

there exists $s' \in Config^\infty$ such that $\alpha_{i,B}(s') = \alpha_{i,B}(s) + r$ and $s \xrightarrow{s'}_{\mathcal{N}_i} s'$. Hence we can conclude $u = s'$. Further since $\|\alpha_{i,B}(s) + r\|_{\mathcal{V}_{i,B}} < B$ we know that $\|\alpha_{i,B}(s')\|_{\mathcal{V}_{i,B}} < B$ which implies that for all $1 \leq i \leq j$ we have $|s'(p_{(j)})| = \alpha_{i,B}(s')(i) < B$ and thus $\|\mathcal{N}_i\| s' < B$. Hence $(s, \alpha_{i,B}(s')) \in \mathcal{B}$ and so we know that \mathcal{B}^{-1} is a simulation.

We can thus conclude that \mathcal{B} is a bisimulation. \square

To establish the relationship between $\mathcal{V}_{i,B}^{\leq}$, $\mathcal{V}_{i,B}$ and \mathcal{N}_i we relabel their transition relations temporarily with a label indicating which rule set was used: \sim for a rule from $\hat{F}_{i,B}$ and ϵ otherwise:

$v \xrightarrow{\sim}_{\mathcal{V}_{i,B}^{\leq}} v'$ if $v \xrightarrow{r}_{\mathcal{V}_{i,B}} v'$, and $v \xrightarrow{\epsilon}_{\mathcal{V}_{i,B}^{\leq}} v'$ if $v \xrightarrow{r}_{\mathcal{V}_{i,B}^{\leq}} v'$ and $r \in \hat{F}_{i,B}^{\leq}$; and for $\mathcal{V}_{i,B}$ and \mathcal{N}_i we label all transitions with \sim .

Proposition 6. The relation $\{(\alpha_{i,B}(s), \alpha_{i,B}(s)) \mid s \in Config^\infty\}$ is a simulation relation for $\mathcal{V}_{i,B}$ and $\mathcal{V}_{i,B}^{\leq}$. The relation $\{(\alpha_{i,B}(s), s') \mid s \leq_{Config} s', s, s' \in Config^\infty, |s(p_{(j)})| = \infty, i < j\}$ is a weak simulation for $\mathcal{V}_{i,B}^{\leq}$ and \mathcal{N}_i .

Proof. That $\{(\alpha_{i,B}(s), \alpha_{i,B}(s)) \mid s \in Config^\infty\}$ is a simulation relation for $\mathcal{V}_{i,B}$ and $\mathcal{V}_{i,B}^{\leq}$ is obvious, since if $v \xrightarrow{\sim}_{\mathcal{V}_{i,B}} v'$ then $v \xrightarrow{\sim}_{\mathcal{V}_{i,B}^{\leq}} v'$.

For the other direction let $\mathcal{W} = \{(\alpha_{i,B}(s), s') \mid s \leq_{Config} s', s, s' \in Config^\infty, |s(p_{(j)})| = \infty, i < j\}$. Suppose $(\alpha_{i,B}(s), \alpha_{i,B}(s')) \in \mathcal{W}$ then we know that $s \leq_{Config} s'$. Suppose that $\alpha_{i,B}(s) \xrightarrow{\sim}_{\mathcal{V}_{i,B}^{\leq}} v$ then we may split this transition up $\alpha_{i,B}(s) \xrightarrow{\epsilon}_{\mathcal{V}_{i,B}^{\leq}} v_1 \xrightarrow{\epsilon}_{\mathcal{V}_{i,B}^{\leq}} \dots \xrightarrow{\epsilon}_{\mathcal{V}_{i,B}^{\leq}} v_{n-1} \xrightarrow{\sim}_{\mathcal{V}_{i,B}^{\leq}} v_n \xrightarrow{\epsilon}_{\mathcal{V}_{i,B}^{\leq}} \dots \xrightarrow{\epsilon}_{\mathcal{V}_{i,B}^{\leq}} v$.

Let us prove a Lemma:

Lemma. If $\alpha_{i,B}(s_0) \xrightarrow{r}_{\mathcal{V}_{i,B}} \alpha_{i,B}(s_0) + r$ with $r \in \hat{F}_{i,B}^{\leq}$ then there exists $s_1 \in Config^\infty$ such that $\alpha_{i,B}(s_1) = \alpha_{i,B}(s_0) + r$ and $s_1 \leq_{Config} s_0$.

Proof. Since $r \in \hat{F}_{i,B}^{\leq}$ we know $r = \alpha_{i,B}(M_{p,m'}) - \alpha_{i,B}(M_{p,m})$ for some $m, m' \in \mathcal{M}^{col}$, such that both $\|m\|_{col}, \|m'\|_{col} \leq B$ and $m >_{\mathcal{M}^{col}} m'$. Suppose $p = p'_{(j_{n_{col}})}$ and $j_0, \dots, j_{n_{col}-1} \in \mathbb{N}^{\leq B}$ such that $m \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ then by definition

$$r \left(i + \sum_{k=0}^{n_{col}} j_k (B+1)^k \right) = -1,$$

since $\alpha_{i,B}(s_0) \rightarrow_{\mathcal{V}_{i,B}^{\leq}} \alpha_{i,B}(s_0) + r$. we must have that

$$\alpha_{i,B}(s_0)(\theta) + r(\theta) \geq 0$$

where $\theta = i + \sum_{k=0}^{n_{col}} j_k (B+1)^k$. Hence we can deduce that

$$\sum_{m^{col} \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})} |s_0(p'_{(j_{n_{col}})})(m^{col})| \geq 1$$

Thus there exists a complex token $m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ such that $|s_0(p'_{(j_{n_{col}})})(m_0)| \geq 1$.

Since $\|m\|_{col} \leq B$ we know that

$$\begin{aligned} m(p_{(k)}^{col}) &= \min(m(p_{(k)}^{col}), B) = j_{k-1} \\ &= \min(m_0(p_{(k)}^{col}), B) \leq m_0(p_{(k)}^{col}) \end{aligned}$$

for all $1 \leq k \leq n_{col}$ and hence $m' <_{\mathcal{M}^{col}} m \leq_{\mathcal{M}^{col}} m_0$.

Clearly $s_0 \oplus s_{p,m_0} \in Config^\infty$ and hence Lemma 16 gives

$$\begin{aligned} \alpha_{i,B}(s_0 \oplus s_{p,m'} \ominus s_{p,m_0}) &= \alpha_{i,B}(s_0) + \alpha_{i,B}(s_{p,m'}) \\ &\quad - \alpha_{i,B}(s_{p,m_0}) \end{aligned}$$

since $m, m_0 \in \gamma_B^{col}(j_0, \dots, j_{n_{col}-1})$ Lemma 17 gives

$$\begin{aligned} \alpha_{i,B}(s_0 \oplus s_{p,m'} \ominus s_{p,m_0}) &= \alpha_{i,B}(s_0) + \alpha_{i,B}(s_{p,m'}) \\ &\quad - \alpha_{i,B}(s_{p,m}) \\ &= \alpha_{i,B}(s_0) + r \end{aligned}$$

It is easy to see that $s_1 := s_0 \oplus s_{p,m'} \ominus s_{p,m_0} \leq_{Config} s$ which concludes the proof of the Lemma. \square

The Lemma allows us to conclude that there are $s_j \in Config^\infty$ such that $v_j = \alpha_{i,B}(s_j)$ for all $j \in \langle n-1 \rangle$, $|s_j(p_{(j')})| = \infty$ for all $i < j'$ and $s_j \leq_{Config} s_{j-1} \leq_{Config} s$ for $1 < j < n$. Inspecting the transition $\alpha_{i,B}(s_{n-1}) \xrightarrow{\nu_{i,B}^\leq} v_n$ then due to the labelling we know that $\alpha_{i,B}(s_{n-1}) \rightarrow_{\nu_{i,B}} v_n$. Lemma 19 then yields that there is a $s'' \in Config^\infty$ such that $s_{n-1} \rightarrow_{\mathcal{N}_i} s''$ and $v_n = \alpha_{i,B}(s'')$. Since $s \leq_{Config} s'$ and \mathcal{N}_i is a WSTS we obtain $\exists s''' \in Config^\infty$ such that $s' \xrightarrow{\mathcal{N}_i} s'''$ and $s'' \leq_{Config} s'''$. Applying the above lemma further we can see that $v = \alpha_{i,B}(s''')$ for some $s''' \leq_{Config} s''$ which implies $s''''' \leq_{Config} s'''$ and hence that $(\alpha_{i,B}(s'''), s''') \in \mathcal{W}$ which is what we wanted to prove.

Thus \mathcal{W} is a weak simulation. \square

Corollary 4. Let $1 \leq i \leq n_s$, $B \in \mathbb{N}$. Suppose $s, s' \in Config^\infty$ such that there exists a covering path from s for s' in $\mathbb{B}_B[\mathcal{N}_i]$ then there exists a covering path from $\alpha_{i,B}(s)$ for $\alpha_{i,B}(s')$ in $\mathbb{B}_B[\mathcal{V}_{i,B}^\leq]$.

Proof. Let s be a covering path for s' from s in $\mathbb{B}_B[\mathcal{N}_i]$. Then Proposition 5 yields a bisimulation for $\mathbb{B}_B[\mathcal{N}_i]$ and $\mathbb{B}_B[\mathcal{V}_{i,B}]$. Hence $\alpha_{i,B}(s) = \alpha_{i,B}(s(1)) \cdots \alpha_{i,B}(s(|s|))$ is a path in $\mathbb{B}_B[\mathcal{V}_{i,B}]$ from $\alpha_{i,B}(s)$ to $\alpha_{i,B}(s(|s|))$.

It is immediate that $\alpha_{i,B}(s)$ is also a path in $\mathbb{B}_B[\mathcal{V}_{i,B}^\leq]$.

Since s is a covering path for s' in $\mathbb{B}_B[\mathcal{N}]$ we have $s' \leq_{Config} s(|s|)$ and hence for all $1 \leq j \leq i$ we have $|s(|s|)(p_{(j)})| \geq |s'(p_{(j)})|$, further for all $1 \leq j \leq n_c$ we have $s(s)(p_{(j)}) \leq_{\mathbb{M}[\mathcal{M}^{col}]} s'(p_{(j)})$. Hence we know that for all $1 \leq j \leq n_c$

$$s'(p_{(j)}) = [m_1^j, \dots, m_{k_j}^j] \quad \text{and} \quad s(s)(p_{(j)}) = [m_1^{j'}, \dots, m_{k_j}^{j'}]$$

such that for all $1 \leq j \leq n_c$ and $1 \leq j' \leq k_j$ we have $m_{j'}^{j'} \leq_{\mathcal{M}^{col}} m_{j'}^{j'}$ (where we wlog assume the injection $h(j') = j'$).

We will now extend $\alpha_{i,B}(s)$ to a new path $\alpha_{i,B}(s)s'$ so that $s'(|s'|)$ covers $\alpha_{i,B}(s')$ in $\mathcal{V}_{i,B}$. This will be achieved by using rules in $\widehat{F}_{i,B}^\leq$ that will allow us to swap the counter abstraction representation of the complex tokens $m_{j'}^{j'}$ for $m_{j'}^j$.

Let us write $t_0 = s(s)$ using the rules in $\widehat{F}_{i,B}^\leq$

$$\left(\alpha_{i,B}(M_{p(1), m_{j'}^j}) - \alpha_{i,B}(M_{p(1), m_{j'}^{j'}}), \alpha_{i,B}(M_{p(1), m_{j'}^{j'}}) \right)$$

for $1 \leq j \leq k_1$ we can see using Lemma 16 that

$$\begin{aligned} \alpha_{i,B}(t_0) &\rightarrow_{\mathcal{V}_{i,B}^\leq} \alpha_{i,B}(t_0[p(1) \mapsto (s(p(1)) \ominus [m_1^1] \oplus [m_1^1])]) \\ &\rightarrow_{\mathcal{V}_{i,B}^\leq} \cdots \\ &\rightarrow_{\mathcal{V}_{i,B}^\leq} \alpha_{i,B}(t_0[p(1) \mapsto (s(p(1)) \ominus [m_1^1, \dots, m_{k_1}^1] \\ &\quad \oplus [m_{j'}^1, \dots, m_{k_1}^1])]) \end{aligned}$$

Clearly

$$\begin{aligned} t_0[p(1) \mapsto (s(p(1)) \ominus [m_1^1, \dots, m_{k_1}^1] \oplus [m_{j'}^1, \dots, m_{k_1}^1])]) \\ = t_0[p(1) \mapsto [m_1^1, \dots, m_{k_1}^1, m_{k_1+1}^1, \dots, m_{k_1'}^1]] =: t_1 \end{aligned}$$

and we note that $s' \leq_{\mathbb{B}_B[\mathcal{N}]} t_1$. We can simply carry on in this fashion and obtain

$$\alpha_{i,B}(t_1) \rightarrow_{\mathcal{V}_{i,B}^\leq}^* \alpha_{i,B}(t_2) \rightarrow_{\mathcal{V}_{i,B}^\leq}^* \cdots \rightarrow_{\mathcal{V}_{i,B}^\leq}^* \alpha_{i,B}(t_{n_c})$$

and for all $1 \leq j \leq n_c$ we have

$$t_j = t_{j-1} [p(j) \mapsto [m_1^j, \dots, m_{k_j}^j, m_{k_j+1}^j, \dots, m_{k_j'}^j]]$$

By construction we know that $\alpha_{i,B}(s(|s|)) \rightarrow_{\mathcal{V}_{i,B}^\leq}^* \alpha_{i,B}(t_{n_c})$ and so there is a path s' from $\alpha_{i,B}(s(|s|))$ to $\alpha_{i,B}(t_{n_c})$ and we further have for all $p \in \mathcal{P}^c$, $m \in \mathcal{M}^{col}$, $|s'(p)(m)| \leq |t_{n_c}(p)(m)|$, for all $p \in \mathcal{P}^s$, $|s'(p)| \leq |t_{n_c}(p)|$.

Thus it is routine to check that $\alpha_{i,B}(s') \leq_{\mathbb{N}^{\hat{\alpha}_{i,B}}} \alpha_{i,B}(t_{n_c})$ and thus $\alpha_{i,B}(s)s'$ is a covering path for $\alpha_{i,B}(s')$ from $\alpha_{i,B}(s)$ and we note that clearly $\|\alpha_{i,B}(s)s'\|_{\mathcal{V}_{i,B}}^* < B$.

Hence we can conclude that $dist_{\mathbb{B}_B[\mathcal{N}]}(s, s') > 0 \implies dist_{\mathbb{B}_B[\mathcal{V}_{i,B}^\leq]}(\alpha_{i,B}(s), \alpha_{i,B}(s')) > 0$. \square

As a consequence we can reason about covering distance on $\mathcal{V}_{i,B}^\leq$ rather than \mathcal{N}_i for configuration pairs in $S_{i,B}$.

Corollary 5. Let $i \leq n_s$, $B \in \mathbb{N}$. For all $(s, s') \in S_{i,B}$ we have $dist_{\mathcal{N}_i}(s, s') \leq dist_{\mathcal{V}_{i,B}^\leq}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$.

Proof. Since $(s, s') \in S_{i,B}$ we know that there exists a covering path s in \mathcal{N}_i from s for s' with norm $\|s\|_{\mathcal{N}_i}^* < B$, i.e. s is a path in $\mathbb{B}_B[\mathcal{N}_i]$.

Corollary 4 then yields that there exists a covering path s_0 from $\alpha_{i,B}(s)$ for $\alpha_{i,B}(s')$ in $\mathbb{B}_B[\mathcal{V}_{i,B}^\leq]$

In particular s_0 is a path in $\mathcal{V}_{i,B}^\leq$ from $\alpha_{i,B}(s)$ that covers $\alpha_{i,B}(s')$. Hence we can find a path s_1 in $\mathcal{V}_{i,B}^\leq$ from $\alpha_{i,B}(s)$ that covers $\alpha_{i,B}(s')$ such that $|s_1| = dist_{\mathcal{V}_{i,B}^\leq}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$.

Proposition 6 then gives us that \mathcal{W} is a weak simulation for $\mathcal{V}_{i,B}^\leq$ and \mathcal{N}_i , and since clearly $(\alpha_{i,B}(s), s) \in \mathcal{W}$ an easy induction on the length of s_1 gives us a path s' in \mathcal{N}_i such that $|s'| \leq |s_1| = dist_{\mathcal{V}_{i,B}^\leq}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$ and $(\alpha_{i,B}(s''), s'(|s'|)) \in \mathcal{W}$ where $s_1(|s_1|) = \alpha_{i,B}(s'')$ which by definition means $s'' \leq_{Config} s(|s|)$

Since $\alpha_{i,B}(s'') \geq_{\mathbb{N}^{\hat{d}_{i,B}}} \alpha_{i,B}(s')$ we know that for all $1 \leq j \leq n_S$, $|s''(p(j))| \geq |s'(p(j))|$. In order to compare the complex places let us fix $p \in \mathcal{P}^C$ and enumerate all elements of $(\mathbb{N}^{\leq B})^{n_{col}}$ by $\mathbf{j}_1, \dots, \mathbf{j}_N$ where $N = |(\mathbb{N}^{\leq B})^{n_{col}}|$. Since $\alpha_{i,B}(s'') \geq_{\mathbb{N}^{\hat{d}_{i,B}}} \alpha_{i,B}(s')$ we know that for all $1 \leq k \leq N$

$$\sum_{m^{\mathbf{j}_k} \in \Xi} s'(p)(m^{\mathbf{j}_k}) \leq \sum_{m^{\mathbf{j}_k} \in \Xi} s''(p)(m^{\mathbf{j}_k})$$

where we abbreviate $\Xi = \gamma_B^{col}(\mathbf{j}_k(1), \dots, \mathbf{j}_k(n_{col}))$. Hence we can conclude that

$$s'(p) = [m_1^{\mathbf{j}_1}, \dots, m_{n_1}^{\mathbf{j}_1}, \dots, m_1^{\mathbf{j}_N}, \dots, m_{n_N}^{\mathbf{j}_N}]$$

$$s''(p) = [m_1^{\mathbf{j}_1}, \dots, m_{n_1}^{\mathbf{j}_1}, \dots, m_1^{\mathbf{j}_N}, \dots, m_{n_N}^{\mathbf{j}_N}]$$

where for all $1 \leq k \leq N$ and $1 \leq k' \leq n_k$, $m_{k'}^{\mathbf{j}_k}, m_{k'}^{\mathbf{j}_{k'}} \in \gamma_B^{col}(\mathbf{j}_k(1), \dots, \mathbf{j}_k(n_{col}))$ and

$$n_k = \sum_{m^{\mathbf{j}_k} \in \Xi} s'(p)(m^{\mathbf{j}_k}), \quad n'_k = \sum_{m^{\mathbf{j}_k} \in \Xi} s''(p)(m^{\mathbf{j}_k})$$

and hence $n_k \leq n'_k$. Let us pair up $m_{k'}^{\mathbf{j}_k}$ and $m_{k'}^{\mathbf{j}_{k'}}$ for all $1 \leq k \leq N$ and $1 \leq k' \leq n_k$. Since $\|s'\|_{\mathcal{N}_{i,C}} \leq B$ we know that $\|m_{k'}^{\mathbf{j}_k}\|_{col} \leq B$ and hence since $m_{k'}^{\mathbf{j}_k}, m_{k'}^{\mathbf{j}_{k'}} \in \gamma_B^{col}(\mathbf{j}_k(1), \dots, \mathbf{j}_k(n_{col}))$ we can deduce that $m_{k'}^{\mathbf{j}_k} \leq_{\mathcal{M}^{col}} m_{k'}^{\mathbf{j}_{k'}}$. Hence we can deduce that $s'(p) \leq_{\mathbb{M}[\mathcal{M}^{col}]} s''(p)$ for all $p \in \mathcal{P}^C$ and thus $s' \leq_{Config} s''$. Since $s'' \leq_{Config} s'(|s'|)$ we can conclude that $s' \leq_{Config} s'(|s'|)$.

Hence s' is a path in \mathcal{N}_i from s that covers s' and $|s'| \leq dist_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$ from which we can conclude that $dist_{\mathcal{N}_i}(s, s') \leq dist_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$. \square

Theorem 5. For all $i \leq n_S$, $B \in \mathbb{N}$ there exists a Petri net $\mathcal{V}_{i,B} = (d_{i,B}, F_{i,B})$ and a function $\alpha_{i,B}$ such that

- (A1) $d_{i,B} \leq i + n_C \times (B+1)^{n_{col}}$,
- (A2) $R \geq \max\{r(i) \mid r \in F_{i,B}\}$, and
- (A3) for all $s, s' \in S_{i,B}$: $R' \geq \max_{j \in \langle d_{i,B} \rangle}(\alpha_{i,B}(s)(j))$,
- (A4) $dist_{\mathcal{N}_i}(s, s') \leq dist_{\mathcal{V}_{i,B}}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$.

Proof. Taking $\mathcal{V}_{i,B} = \mathcal{V}_{i,B}^{\leq}$ it is easy to see that (A1)–(A3) hold. (A4) follows from Corollary 5. \square

2) Proofs of Section IV-B:

Corollary 6. Let $1 \leq i \leq n_S$, $B \in \mathbb{N}$. Then

$$d_{\mathcal{N}_i}(S_{(i,B)}) \leq \max \left\{ \rho_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s')) : \|s'\|_{\mathcal{N}_{i,C}} \leq B \right\}$$

Proof. Let $(s, s') \in S_{(i,B)}$ then Corollary 5 gives that

$$dist_{\mathcal{N}_i}(s, s') \leq dist_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s), \alpha_{i,B}(s'))$$

$$\leq \rho_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s'))$$

because $\alpha_{i,B}(s) \in \mathbb{N}^{\hat{d}_{i,B}}$. Taking max over $S_{(i,B)}$:

$$d_{\mathcal{N}_i}(S_{(i,B)}) = \max \left\{ dist_{\mathcal{N}_i}(s, s') \mid (s, s') \in S_{(i,B)} \right\}$$

$$\leq \max \left\{ \rho_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s')) \mid (s, s') \in S_{(i,B)} \right\}$$

$$\leq \max \left\{ \rho_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s')) : \|s'\|_{\mathcal{N}_{i,C}} < B \right\}$$

which concludes the proof. \square

Corollary 2. Let $i \leq n_S$, $B \in \mathbb{N}$. Then

$$d_{\mathcal{N}_i}(S_{(i,B)}) \leq \max \left\{ \rho_{\mathcal{V}_{i,B}}(\alpha_{i,B}(s')) : \|s'\|_{\mathcal{N}_{i,C}} \leq B \right\}$$

$$\leq (6 \max \{R, B, 1\} \max \{R', B\})^{(\hat{d}_{i,B}+1)!}.$$

Proof. Let s be such that $\|s\|_{\mathcal{N}_{i,C}} \leq B$, then Lemma 6 [3, Lemma 12] applies to give us $\rho_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s)) \leq (6R_{\mathcal{V}_{i,B}^{\leq}} R'_{\mathcal{V}_{i,B}^{\leq}})^{(\hat{d}_{i,B}+1)!}$.

Further it is easy to see that $R_{\mathcal{V}_{i,B}^{\leq}} \leq \max \{R, B, 1\}$ and $R'_{\mathcal{V}_{i,B}^{\leq}} \leq \max \{R', B\}$ and hence we obtain $\rho_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s)) \leq (6 \max \{R, B, 1\} \max \{R', B\})^{(\hat{d}_{i,B}+1)!}$. Invoking Corollary 6 yields:

$$d_{\mathcal{N}_i}(S_{(i,B)}) \leq \max \left\{ \rho_{\mathcal{V}_{i,B}^{\leq}}(\alpha_{i,B}(s')) : \|s'\|_{\mathcal{N}_{i,C}} \leq B \right\}$$

$$\leq \max_{\|s'\|_{\mathcal{N}_{i,C}} \leq B} ((6 \max \{R, B, 1\} \max \{R', B\})^{(\hat{d}_{i,B}+1)!})$$

$$\leq (6 \max \{R, B, 1\} \max \{R', B\})^{(\hat{d}_{i,B}+1)!}$$

where the last inequality is justified since the argument of max does not depend on s' only on B . \square

Theorem 6. Let us write slog, super-logarithm, for the inverse of $2 \uparrow (-)$, i.e., $n = 2 \uparrow \text{slog}(n)$. Then for all $1 \leq i \leq n_S$:

- (i) $\rho_{\mathcal{N}_0}(s_{cov}) \leq 2 \uparrow (2 \text{slog}(48n_C n_S n_{col} R'))$
- (ii) $\rho_{\mathcal{N}_{i+1}}(s_{cov}) \leq 2^{2^{((\max\{\rho_{\mathcal{N}_i}(s_{cov}), 2\})^{48n_{col} n_S n_C R'})}}$ and
- (iii) $\rho_{\mathcal{N}}(s_{cov}) \leq 2 \uparrow (2n_S + 2 \text{slog}(48(n_S + 1)n_{col} n_S n_C R'))$.

Proof. For claim (i) applying Proposition 4 yields that

$$\rho_{\mathcal{N}_0}(s_{cov}) \leq d_{\mathcal{N}_0}(S_{(0,R')})$$

Applying Corollary 2 then gives us:

$$\rho_{\mathcal{N}_0}(s_{cov}) \leq (6 \max \{R, R', 1\} R')^{(\hat{d}_{0,R'}+1)!}$$

$$+ \rho_{\mathcal{N}_i}(s_{cov})$$

Clearly $R' = \max \{R, R', 1\}$. Hence

$$\rho_{\mathcal{N}_0}(s_{cov}) \leq (6R'^2)^{(\hat{d}_{0,R'}+1)!}$$

$$\leq 2^{\sum_{k=1}^{\hat{d}_{0,R'}+6+R'} \log_2(k)}$$

Further

$$\log_2(6R'^2)^{(\hat{d}_{0,R'}+1)!} + 1$$

$$\leq (4 + 2 \log_2(R'))^{(\hat{d}_{0,R'}+1)!}$$

$$\leq (\hat{d}_{0,R'} + 6 + R')!$$

since $4 + 2 \log_2(R') \leq \hat{d}_{0,R'} + 6 + R'$, and hence

$$\log_2(6R'^2)^{(\hat{d}_{0,R'}+1)!} + 1$$

$$\leq 2^{\sum_{k=1}^{\hat{d}_{0,R'}+6+R'} \log_2(k)}$$

$$\leq 2^{(\hat{d}_{0,R'}+6+R')^2}$$

$$\begin{aligned}
&= 2^{((R'+1)((n_c+1)(R'+1)^{n_{col}-1}-1)+6+R')^2} \\
&= 2^{(((n_c+1)(R'+1)^{n_{col}})+5)^2}
\end{aligned}$$

And

$$\begin{aligned}
&2 \log_2(((n_c+1)(R'+1)^{n_{col}})+5) \\
&\leq 2(\log_2(n_c+1) + n_{col} \log_2(R'+1) + \log_2(5)) \\
&\leq 2n_c + 2n_{col}R' + 6 \leq 48n_cn_sn_{col}R'
\end{aligned}$$

Putting it all together gives

$$\rho_{\mathcal{N}_0}(s_{\text{cov}}) \leq 2 \uparrow (2 \log(48n_cn_sn_{col}R'))$$

For the second claim we know from Proposition 4 that

$$\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) \leq d_{\mathcal{N}_{i+1}}(S_{(i+1, B_i)}) + \rho_{\mathcal{N}_i}(s_{\text{cov}}),$$

where $B_i = R' \times \rho_{\mathcal{N}_i}(s_{\text{cov}})$. Corollary 2 then tells us that

$$\begin{aligned}
\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) &\leq (6 \max\{R, B, 1\} \max\{R', B\})^{\widehat{d}_{i, B+1}!} \\
&\quad + \rho_{\mathcal{N}_i}(s_{\text{cov}})
\end{aligned}$$

Then since we know $B_i + 1 \geq \max\{R, R', 1\}$ the above implies

$$\begin{aligned}
\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) &\leq (6(B_i + 1)^2)^{\widehat{d}_{i+1, B_i+1}!} + \rho_{\mathcal{N}_i}(s_{\text{cov}}) \\
&\leq 2^{\log_2(6(B_i+1)^2)(\widehat{d}_{i+1, B_i+1}!)} + 2^{\log_2(\rho_{\mathcal{N}_i}(s_{\text{cov}}))} \\
&\leq 2^{\log_2(6(B_i+1)^2)(\widehat{d}_{i+1, B_i+1}!)+1}
\end{aligned}$$

where the last line is justified since the inequality $\log_2(6(B_i + 1)^2) \geq \log_2(\rho_{\mathcal{N}_i}(s_{\text{cov}}))$ holds. Further

$$\begin{aligned}
&\log_2(6(B_i + 1)^2)(\widehat{d}_{i+1, B_i+1}!)+1 \\
&\leq (4 + 2 \log_2(B_i + 1))(\widehat{d}_{i+1, B_i+1}!)+1 \\
&\leq (\widehat{d}_{i+1, B_i+1} + 6 + B_i)!
\end{aligned}$$

since $4 + 2 \log_2(B_i + 1) \leq \widehat{d}_{i+1, B_i+1} + 6 + B_i$, and hence

$$\begin{aligned}
&\log_2(6(B_i + 1)^2)(\widehat{d}_{i+1, B_i+1}!)+1 \\
&\leq 2^{\sum_{k=1}^{\widehat{d}_{i+1, B_i+1} + 6 + B_i} \log_2(k)} \\
&\leq 2^{(\widehat{d}_{i+1, B_i+1} + 6 + B_i)^2}
\end{aligned}$$

Expanding \widehat{d}_{i+1, B_i} then gives

$$\begin{aligned}
\widehat{d}_{i+1, B_i} + 6 + B_i &= i + (B_i + 1) \left((n_c + 1) (B_i + 1)^{n_{col}-1} - 1 \right) \\
&\quad + 6 + B_i + 1 \\
&= i + (n_c + 1) (B_i + 1)^{n_{col}} + 6 \\
&\leq 2^{\log_2(n_s)} + 2^{\log_2(n_c+1)} (B_i + 2)^{n_{col}} + 2^3 \\
&\leq 2^{3+\log_2(n_s)+\log_2(n_c+1)} (B_i + 2)^{n_{col}} \\
&\leq (B_i + 2)^{n_{col}+3+\log_2(n_s)+\log_2(n_c+1)}
\end{aligned}$$

Expanding B_i

$$\begin{aligned}
B_i + 2 &\leq R' \times \rho_{\mathcal{N}_i}(s_{\text{cov}}) + 2 \\
&\leq (R' + 1) \max\{\rho_{\mathcal{N}_i}(s_{\text{cov}}), 2\} \\
&\leq 2^{\log_2(R'+1)} \max\{\rho_{\mathcal{N}_i}(s_{\text{cov}}), 2\}
\end{aligned}$$

$$\leq (\max\{\rho_{\mathcal{N}_i}(s_{\text{cov}}), 2\})^{\log_2(R'+1)+1}$$

Putting it all together

$$\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) \leq 2^2 \left(\left((\max\{\rho_{\mathcal{N}_i}(s_{\text{cov}}), 2\})^{\log_2(R'+1)+1} \right)^{n_{col}+3+\log_2(n_s)+\log_2(n_c+1)} \right)^2.$$

And we can see

$$\begin{aligned}
&2(\log_2(R'+1) + 1)(n_{col} + 3 + \log_2(n_s) + \log_2(n_c + 1)) \\
&\leq 2(R' + 1)(3 + n_{col} + n_s + n_c) \\
&\leq 48n_{col}n_sn_cR'
\end{aligned}$$

since $n_{col}, n_s, n_c, R' \geq 1$ and hence we can conclude

$$\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) \leq 2^2 \left((\max\{\rho_{\mathcal{N}_i}(s_{\text{cov}}), 2\})^{48n_{col}n_sn_cR'} \right).$$

We can prove claim (iii) by a simple induction. The base case is given by (i). The inductive step is:

$$\begin{aligned}
\rho_{\mathcal{N}_{i+1}}(s_{\text{cov}}) &\leq 2^2 \left(\max\{\rho_{\mathcal{B}, [\mathcal{N}_i]2}\} \right)^{48n_{col}n_sn_cR'} \\
&\leq 2^{2 \uparrow (2 \cdot i + 2 \log(48(i+1)n_{col}n_sn_cR'))} 48n_{col}n_sn_cR' \\
&\leq 2 \uparrow (2 \cdot (i+1) + 2 \log(48(i+2)n_{col}n_sn_cR'))
\end{aligned}$$

which concludes the proof. \square

Corollary 3. Coverability for NNCTs is decidable and in TOWER.

Proof. Let \mathcal{N} be an NNCT with n_s places, n_c complex places, n_{col} colours and rules \mathcal{R} and a coverability query Q giving rise to a bound R' and

$$B(n_s, n_c, n_{col}, R') = 2 \uparrow (2n_s + 2 \log(48(n_s + 1)n_{col}n_sn_cR')).$$

Then the theorem above tells us that along a covering path a simple place cannot contain more than $R \cdot B(n_s, n_c, n_{col}, R')$ tokens, a complex tokens cannot have more that $R \cdot B(n_s, n_c, n_{col}, R')$ tokens of a particular colour and there are at most $R \cdot B(n_s, n_c, n_{col}, R')$ complex tokens. Hence along a covering path simple place can be represented using $\log_2(R \cdot B(n_s, n_c, n_{col}, R'))$ bits, the state of complex token can be represented using $n_{col} \log_2(R \cdot B(n_s, n_c, n_{col}, R'))$ bits and hence a complex place may be represented by $n_cn_{col}R \cdot B(n_s, n_c, n_{col}, R') \log_2(R \cdot B(n_s, n_c, n_{col}, R'))$ bits. Hence a non-deterministic Turing machine requiring at most $O(B(n_s, n_c, n_{col}, R'))$ space can decide the coverability problem. Using Savitch's theorem we know there is a deterministic turing machine deciding coverability in $O(B(n_s, n_c, n_{col}, R')^2)$ space and using an exponential to obtain a time bounded turing machine we find that the coverability problem can be decided in time $O(2^{B(n_s, n_c, n_{col}, R')^2})$ and is thus clearly in TOWER. \square

D. Proof of Theorem 7

Theorem 7. *Coverability for a simple query for total-transfer NNCT is TOWER-hard.*

Proof. We can deduce TOWER-hardness by showing that given a deterministic bounded two-counter machine, \mathcal{M} , of size n with counters that are $(2 \uparrow n)$ -bounded we can construct an NNCT $\mathcal{N}_{\mathcal{M}}$ in polynomial-time that weakly bisimulates \mathcal{M} in such a way that we can reduce the halting problem for \mathcal{M} to coverability for a simple query for $\mathcal{N}_{\mathcal{M}}$. The machine \mathcal{M} can use the following operations: $x++$, $x--$, $reset(x)$, $iszero(x)$, $ismax(x)$ for each counter x .

Each simulation state of $\mathcal{N}_{\mathcal{M}}$ will represent a valuation v of $6n + 2$ *active* and *inactive* counters, and n arrays. In addition to the counters x, y of \mathcal{M} the NNCT $\mathcal{N}_{\mathcal{M}}$ will simulate the auxiliary counters $s_i, p_i, p'_i, c_i, c'_i$ and an auxiliary array a_i for each $i \leq n$. Each active counter $d \in \{s_i, p_i, p'_i, c_i, c'_i\} \cup \{x, y \mid i = n\}$ is $(2 \uparrow i)$ -bounded, each inactive counter has an undefined value. For each i the array has length exactly $(2 \uparrow i) + 1$ and carries values $a_i(j) \in \{0, 1, 2\}$ for $0 \leq j \leq 2 \uparrow i$. The NNCT $\mathcal{N}_{\mathcal{M}}$ will have two simple places $p[d]$ and $\bar{p}[d]$ for each counter $d \in \{s_i, p_i, c_i, c'_i\} \cup \{x, y \mid i = n\}$, three complex places $p[a_i]$ and $\bar{p}[a_i]$, $p[aux_i]$ and two colours $p[j_i]$ and $\bar{p}[j_i]$ for each array a_i , and a complex “sink” place $p[disc]$; ζ maps $p[j_i]$ to $p[p'_i]$ and $\bar{p}[j_i]$ to $\bar{p}[p'_i]$, in addition to a (polynomial in n) number of simple places encoding the control of \mathcal{M} and the “internal” control of $\mathcal{N}_{\mathcal{M}}$. Further $\mathcal{N}_{\mathcal{M}}$ ’s transfer rules will all be total, hence $\mathcal{N}_{\mathcal{M}}$ will be a total-transfer NNCT. A valuation v is represented by a configuration s as follows:

- For each i and $d \in \{s_i, p_i, p'_i, c_i, c'_i\} \cup \{x, y \mid i = n\}$ if d is active then there are exactly $v(d)$ \bullet -tokens in $p[d]$ and $2 \uparrow i - v(d)$ \bullet -tokens in $\bar{p}[d]$.
- For each i and $d \in \{s_i, p_i, p'_i, c_i, c'_i\} \cup \{x, y \mid i = n\}$ if d is inactive then both places $p[d], \bar{p}[d]$ are empty.
- For each i the array a_i is represented as follows: for each $0 \leq k \leq 2 \uparrow i$ let $m_{(i,k)}$ be a complex token such that $m_{(i,k)}$ contains exclusively tokens of colours $p[j_i]$ and $\bar{p}[j_i]$: k tokens of colour $p[j_i]$ and $2 \uparrow i - k$ tokens of colour $\bar{p}[j_i]$; then there are exactly $v(a_i(k))$ complex tokens $m_{(i,k)}$ in $p[a_i]$ and $2 - v(a_i(k))$ complex tokens $m_{(i,k)}$ in $\bar{p}[a_i]$.
- For each i the place $p[aux_i]$ is empty.

The question whether \mathcal{M} halts, i.e. whether \mathcal{M} reaches a halting control state from its initial state, can then be answered by performing a coverability query on $\mathcal{N}_{\mathcal{M}}$ where the query marking involves only the simple places of $\mathcal{N}_{\mathcal{M}}$ encoding \mathcal{M} ’s finite control. Assuming that only \mathcal{M} ’s halting control states have no successors, \mathcal{M} ’s halting problem also reduces to the termination problem for $\mathcal{N}_{\mathcal{M}}$. And we can augment $\mathcal{N}_{\mathcal{M}}$ with an additional simple place that is incremented with every transition so that the halting problem for \mathcal{M} reduces to deciding boundedness of $\mathcal{N}_{\mathcal{M}}$.

Let $d \in \{s_i, p_i, p'_i, c_i, c'_i, x, y \mid 1 \leq i \leq n\}$ we will implement operation $d++$ as follows

Add a \bullet -token to $p[d]$

Remove a \bullet -token from $\bar{p}[d]$

Suppose d is $2 \uparrow i$ -bounded. Clearly if configuration s represents valuation v , d is active and $v(d) < 2 \uparrow i$ then there are $v(d)$ \bullet -tokens in $p[d]$ and $2 \uparrow i - v(d)$ \bullet -tokens in $\bar{p}[d]$. Hence there is at least one \bullet -token in $\bar{p}[d]$. Performing the operation $d++$ yields a new configuration s' where there are $v(d) + 1$ \bullet -tokens in $p[d]$ and $2 \uparrow i - (v(d) + 1)$ \bullet -tokens in $\bar{p}[d]$ and all other (non-control) places are unchanged. The configuration s' then represents the valuation $v[d \mapsto v(d) + 1]$. Further if $v(d) = 2 \uparrow i$ we note $\bar{p}[d]$ is empty and thus the simulation of $d++$ blocks at the attempt to remove a token from $\bar{p}[d]$.

We note that we can implement $d--$ by

Remove a \bullet -token from $p[d]$

Add a \bullet -token to $\bar{p}[d]$

Similarly to the reasoning on $d++$ we can see that if configuration s represents valuation v , d is active and $v(d) > 0$ we can successfully simulate $d--$ and obtain a configuration s' that represents the valuation $v[d \mapsto v(d) - 1]$. However if $v(d) = 0$ simulating $d--$ will get stuck.

In addition to the operations \mathcal{M} supports, we will implement further instructions to simplify and improve readability. The NNCT $\mathcal{N}_{\mathcal{M}}$ will simulate *activate*(d), *deactivate*(d) for $d \in \{s_i, p_i, p'_i, c_i, c'_i, x, y \mid 1 \leq i \leq n\}$, and operations *reset*(d), *iszero*(d), *ismax*(d) for $d \in \{p_i, p'_i, c_i, c'_i, x, y \mid 1 \leq i \leq n\}$. Further we implement the operation *ismax&reset*(s_i) and the counter-specialised operations:

isequal(p_i, p'_i), $a_i(p_i)++$, $a_i(p_i)--$, *reset*($a_i(p_i)$),
iszero($a_i(p_i)$), *ismax*($a_i(p_i)$), *activate*(a_i)

for each array a_i and counter p_i . The latter counter operations will be used to implement *ismax&reset*(s_i). All above operations are only guaranteed to succeed if the counter in question is active at the start of the operation.

Counters $s_1, p_1, p'_1, c_1, c'_1$ are 2-bounded so implementing operations on them is trivial. For $i < n$, operations on a_i are simulated using $s_i, p_i, p'_i, c_i, c'_i$ and operations on $s_{i+1}, p_{i+1}, p'_{i+1}, c_{i+1}, c'_{i+1}$ are simulated using operations on p_i, p'_i, c_i, c'_i and a_i .

(i) The following shows how to implement $a_i(p_i)++$ and can only succeed if p_i, p'_i are active.

Move a complex-token from $\bar{p}[a_i]$ to $p[aux_i]$;

deactivate(p'_i);

Eject the contents of a complex-token m in $p[aux_i]$ and its remains into $p[disc]$;

isequal(p_i, p'_i); *reset*(p'_i);

Create an empty complex-token in $p[aux_i]$;

while ($p_i \neq p'_i$) **do**

Inject a $p[j_i]$ -coloured \bullet -token into a complex token in $p[aux_i]$;

p'_i++ ;

while ($\neg(ismax(p'_i))$) **do**

Inject a $\bar{p}[j_i]$ -coloured \bullet -token into a complex token in $p[aux_i]$;

p'_i++ ;

Move a complex-token from $p[aux_i]$ to $p[a_i]$;

reset(p'_i);

Suppose $a_i(p_i)++$ is executed in a configuration s that represents valuation v and p_i, p'_i are active. If $v(a_i(v(p_i))) < 2$ then we know there exists a complex token $m_{(i,v(p_i))}$ in $\bar{p}[a_i]$ and all complex tokens in $\bar{p}[a_i]$ are of the form $m_{(i,k)}$ for some $0 \leq k \leq 2 \uparrow i$. The move of one $m_{(i,k)}$ complex token from $\bar{p}[a_i]$ to $p[aux_i]$ results in $p[aux_i]$ containing just $m_{(i,k)}$, since by assumption $p[aux_i]$ was empty before. After deactivating p'_i we know that both $p[p'_i]$ and $\bar{p}[p'_i]$ are empty. Ejecting the contents of $m_{(i,k)}$ removes $m_{(i,k)}$ from $p[aux_i]$, inserts k \bullet -tokens into $p[p'_i]$ and $(2 \uparrow i) - k$ \bullet -tokens into $\bar{p}[p'_i]$ and places the remaining (now empty) complex token into $p[disc]$. We can see that, disregarding a_i , the configuration we have reached represents a partial valuation v' that sets $v'(p'_i) = k$ and $v'(p_i) = v(p_i)$. After executing $isequal(p_i, p'_i)$ the simulation only succeeds if $k = v(p_i)$. Hence $\bar{p}[a_i]$ now contains $2 - (v(a_i(v(p_i)))) + 1$ complex tokens $m_{(i,v(p_i))}$ and the same number of other complex tokens $m_{(i,k)}$. Clearly the two following while loops carefully inject $p[j_i]$ -coloured \bullet -tokens and $\bar{p}[j_i]$ -coloured \bullet -tokens into the newly created token at $p[aux_i]$ to yield a new $m_{(i,v(p_i))}$ located in $p[aux_i]$ which we move to $p[a_i]$. Thus $p[a_i]$ now contains $v(a_i(v(p_i))) + 1$ complex tokens $m_{(i,v(p_i))}$ and the same number of other complex tokens $m_{(i,k)}$ as before. The configuration s' we have reached thus represents a valuation v'' such that $v''(a_i(j)) = v(a_i(j))$ for all $0 \leq j \leq 2 \uparrow i$ and $j \neq v(p_i)$ and $v''(a_i(v(p_i))) = v(a_i(v(p_i))) + 1$. Otherwise if $v(a_i(v(p_i))) = 2$ then the simulation either blocks while attempting to move a complex token $m_{(i,k)}$ from $\bar{p}[a_i]$ to $p[aux_i]$ or on the execution of $isequal(p_i, p'_i)$ since it is impossible to obtain $k = v(p_i)$.

By swapping $p[a_i]$ and $\bar{p}[a_i]$ in the above we obtain an implementation of $a_i(p_i)--$.

(ii) The following shows how to implement $activate(a_i)$ and can only succeed if p_i and p'_i are active.

```

for  $p_i := 0$  to  $2 \uparrow i$  do
  for  $z := 0$  to  $1$  do
    Create an empty complex-token in  $p[aux_i]$ ;
    while  $(p_i \neq p'_i)$  do
      Inject a  $p[j_i]$ -coloured  $\bullet$ -token into a
      complex token in  $p[aux_i]$ ;
       $p'_i++$ ;
    while  $(\neg(ismax(p'_i)))$  do
      Inject a  $\bar{p}[j_i]$ -coloured  $\bullet$ -token into a
      complex token in  $p[aux_i]$ ;
       $p'_i++$ ;
    Move a complex-token from  $p[aux_i]$  to  $\bar{p}[a_i]$ ;
     $reset(p'_i)$ ;

```

We first note that the interior for-loop simply repeats its body twice and can thus be considered syntactic sugar. Suppose $activate(a_i)$ is invoked in a configuration s such that in s the places $p[a_i]$ and $\bar{p}[a_i]$ are empty. We can then follow our analysis in the second part of $a_i(p_i)++$'s implementation to see that the interior for-loop places two complex tokens $m_{i,k}$ into $\bar{p}[a_i]$ for all $k \in 0, \dots, 2 \uparrow i$. Hence when the outer for-loop terminates we have reached a configuration representing a valuation v such that $v(a_i(k)) = 0$ for all $0 \leq k \leq 2 \uparrow i$.

(iii) The following shows how to implement $iszero(a_i(p_i))$.

```

 $a_i(p_i)++$ ;  $a_i(p_i)++$ ;
 $a_i(p_i)--$ ;  $a_i(p_i)--$ ;

```

Suppose $iszero(a_i(p_i))$ is executed in a configuration s that represents valuation v and p_i, p'_i are active. If $v(a_i(v(p_i))) = 0$ we can clearly execute $a_i(p_i)++$ twice and undo the former by executing $a_i(p_i)--$ twice. We thus end up at a configuration representing v . If however $v(a_i(v(p_i))) > 0$ then either $v(a_i(v(p_i))) = 2$ initially or after the first invocation of $a_i(p_i)++$ and our analysis above assures us that the subsequent invocation of $a_i(p_i)++$ blocks. Hence $iszero(a_i(p_i))$ only succeeds if $v(a_i(v(p_i))) = 0$. We obtain an implementation of $ismax(a_i(p_i))$ by swapping $a_i(p_i)++$ for $a_i(p_i)--$ and vice versa. Analogous reasoning to above then yields that $iszero(a_i(p_i))$ only succeeds if $v(a_i(v(p_i))) = 2$.

(iv) The following shows how to implement $reset(a_i(p_i))$.

```

while  $(a_i(p_i) \neq 0)$  do
   $a_i(p_i)--$ 

```

It is trivial to see that the while loop only terminates once a configuration is reached representing a valuation v that sets $v(a_i(v(p_i))) = 0$. In every iteration $v(a_i(v(p_i)))$ is decremented by 1, so termination is ensured.

(v) The following shows how to implement $ismax \& reset(s_{i+1})$.

```

for  $p_i := 0$  to  $2 \uparrow i$  do
   $reset(a_i(p_i))$ 
  while  $(iszero(a_i(2 \uparrow i)))$  do
     $s_{i+1}--$ ;  $reset(p_i)$ ;  $a_i(p_i)++$ ;
    while  $ismax(a_i(p_i))$  do
       $a_i(p_i)--$ ;  $p_i++$ ;  $a_i(p_i)++$ ;

```

We know that after the for-loop a_i is the array such that $a_i(x) = 0$ for all $0 \leq x \leq 2 \uparrow i$. The array a_i is meant to be binary representation of a number between 0 and $2 \uparrow (i + 1)$. This number is initially 0 and we can see that the outer while loop performs a long addition of 1 for each iteration. If $v(a_i(v(p_i))) = 2$ then $v(p_i)$ is an index representing a carry bit in the long addition computation. Clearly for each number represented by a_i we have performed $s_{i+1}--$. Hence if initially $v(s_{i+1}) = 2 \uparrow (i + 1)$ then after performing $ismax \& reset(s_{i+1})$ it is the case that the resulting valuation $v'(s_{i+1}) = 0$, and if $v(s_{i+1}) < 2 \uparrow (i + 1)$ then after $v(s_{i+1})$ iterations the resulting valuation v' would set $v'(s_{i+1}) = 0$ and a_i would represent the number $v(s_{i+1})$. Since $v(s_{i+1}) < 2 \uparrow (i + 1)$ this implies that $a_i(2 \uparrow i) = 0$ and hence body of the outer while loop is executed again leading to an invocation of $s_{i+1}--$ which will block. Hence $ismax \& reset(s_{i+1})$ will block when executed in a configuration representing v such that $v(s_{i+1}) < 2 \uparrow (i + 1)$.

We modify the implementation of $ismax \& reset(s_{i+1})$ by replacing $s_{i+1}--$ by *Add a \bullet -token to $\bar{p}[d]$* we obtain an implementation of $activate(d)$. If d is inactive then $p[d]$ and $\bar{p}[d]$ are empty. By an analogous argument to above we can see that invoking $activate(d)$ will add $2 \uparrow (i + 1)$ \bullet -token to $\bar{p}[d]$ yielding a configuration where d is active and is valued 0. We can similarly obtain an implementation for $deactivate(d)$. First modify $ismax \& reset(s_{i+1})$ by replacing $s_{i+1}--$ by *Remove*

a \bullet -token from $\bar{p}[d]$ which gives us an implementation of an intermediate operation $deactivate'(d)$. If d is active and valued 0 then $deactivate'(d)$ clearly succeeds and removes all tokens from $\bar{p}[d]$. Hence we can implement $deactivate(d)$ by $reset(d); deactivate'(d)$; which succeeds if d is active.

(vi) The following shows how to implement $iszero(d)$ for $d \in \{p_{i+1}, p'_{i+1}, c_{i+1}, c'_{i+1}\} \cup \{x, y \mid i+1 = n\}$ if s_i is 0.

```

while (*) do
   $d++$ ;  $s_{i+1}++$ ;
   $ismax \& reset(s_{i+1})$ 
while (*) do
   $d--$ ;  $s_{i+1}++$ ;
   $ismax \& reset(s_{i+1})$ 

```

Suppose $iszero(d)$ is started in a configuration s_0 that represents valuation v_0 such that $v_0(s_i) = 0$. After the first while-loop non-deterministically terminates, just before invoking $ismax \& reset(s_{i+1})$ we reach a configuration that represents a valuation v such that $v(s_{i+1}) = k$ and $v(d) = v_0(d) + k$. Since neither $d++$ nor $s_{i+1}++$ blocked we know that $0 \leq k \leq 2 \uparrow (i+1) - v_0(d)$. In fact for all $0 \leq k \leq 2 \uparrow (i+1) - v_0(d)$ it is possible to reach such a configuration where $v(s_{i+1}) = k$ and $v(d) = v_0(d) + k$. Invoking $ismax \& reset(s_{i+1})$ only succeeds if $k = 2 \uparrow (i+1)$. Thus all other configurations where $k < 2 \uparrow (i+1)$ $ismax \& reset(s_{i+1})$ blocks. Hence if $v_0(d) > 0$ it will be the case that $k < 2 \uparrow (i+1)$ and thus $ismax \& reset(s_{i+1})$ blocks. If $v_0(d) = 0$ then there is one configuration that we can reach which represents a valuation v_1 such that $v_1(s_{i+1}) = 2 \uparrow (i+1)$, $v_1(d) = v_0(d) + 2 \uparrow (i+1) = 2 \uparrow (i+1)$ and we can successfully invoke $ismax \& reset(s_{i+1})$ to yield a new configuration representing a valuation v_2 $v_2(s_{i+1}) = 0$, $v_2(d) = 2 \uparrow (i+1)$. The second while loop then again non-deterministically decrements d and increments s_{i+1} in the same fashion as above. Again similar reasoning to above yields that there is only one configuration that we can reach that represents a valuation v_3 which values $v_3(s_{i+1}) = 2 \uparrow (i+1)$ and consequently $v_3(d) = 0$ and which guarantees $ismax \& reset(s_{i+1})$ succeeds to yield a configuration that represents a valuation v_4 such that $v_4(s_{i+1}) = v_4(d) = 0$. Hence $iszero(d)$ succeeds only if started in a configuration that values d as 0.

We can modify the above to obtain an implementation $ismax(d)$. We simply swap the first for the second while loop and analogous reasoning to $iszero(d)$ then yields that $ismax(d)$ succeeds only when d is valued at $2 \uparrow (i+1)$ and s_{i+1} at 0.

(vii) The following shows how to implement $isequal(p_i, p'_i)$ for $d \in \{p_i, p'_i, c_i, c'_i\} \cup \{x, y \mid i = n\}$ if both p_i and p'_i are active and c_i is zero.

```

while ( $p_i \neq 0$  or  $p'_i \neq 0$ ) do
   $p_i--$ ;  $p'_i--$ ;  $c_i++$ ;
while ( $c_i \neq 0$ ) do
   $p_i++$ ;  $p'_i++$ ;  $c_i--$ ;

```

If we invoke $isequal(p_i, p'_i)$ in a configuration that represents a valuation v such that $v(p_i) = k$, $v(p'_i) = k'$ and $v(c_i) = 0$ then for each iteration of the first while loop we obtain a configuration representing a valuation v' such that

$v'(p_i) = k - v'(c_i)$, $v'(p'_i) = k' - v'(c_i)$. The inequality $v'(c_i) \leq \min(k, k')$ must hold at the end of each iteration as otherwise either p_i-- or p'_i-- would have blocked. Since the loop-guard is false only if both $v'(p_i) = 0$ and $v'(p'_i) = 0$ we can see that the while loop only terminates successfully once $v'(c_i) = k = k'$ i.e. if initially $v(p_i) = v(p'_i)$. Otherwise the loop guard would never be false and thus at some point the invocation of p_i-- or p'_i-- blocks. Thus after the successful breaking out of the while loop $v'(c_i) = v(p_i) = v(p'_i)$ and it is easy to see that the second while loop simply copies the contents of c_i back to p_i and p'_i . Hence invoking $isequal(p_i, p'_i)$ only succeeds if started in a configuration where $v(p_i) = v(p'_i)$ and $v(c_i) = 0$.

(viii) The following shows how to implement $reset(d)$ for $d \in \{p_i, p'_i, c_i, c'_i\} \cup \{x, y \mid i = n\}$.

```

while ( $d \neq 0$ ) do
   $d--$ 

```

As in the implementation of $reset(a_i(p_i))$ the while-loop simply decrements d until d has value 0. Hence $reset(d)$ works as expected.

In order to set up a configuration that simulates the initial configuration of \mathcal{M} we set up $\mathcal{N}_{\mathcal{M}}$ where all non-control places are empty and $\mathcal{N}_{\mathcal{M}}$'s finite control initiates the execution of

```

 $activate(s_1); activate(p_1); activate(p'_1);$ 
 $activate(c_1); activate(c'_1); activate(a_1);$ 
...
 $activate(s_{n-1}); activate(p_{n-1}); activate(p'_{n-1});$ 
 $activate(c_{n-1}); activate(c'_{n-1}); activate(a_{n-1});$ 
 $activate(s_n); activate(p_n); activate(p'_n); activate(c_n);$ 
 $activate(c'_n); activate(x); activate(y);$ 

```

After executing these operations clearly we reach a configuration s in which all counters are active and which represents a valuation v such that $v(d) = 0$ for all counters d and $v(a_i(k)) = 0$ for all $0 \leq k \leq 2 \uparrow i$. □